IMPROVED PREDICTION OF BIODEGRADATION PATHWAYS: VISUALIZATION AND PERFORMANCE

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

JUNFENG GAO

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

LYNDA B.M. ELLIS

FEBRUARY 2011

Acknowledgements

I would like to thank all of those people who helped make this dissertation possible.

First, I wish to thank my advisor and mentor, Dr. Lynda Ellis for all her guidance, encouragement, support, and patience. Her sincere interests in science and education have been a great inspiration to me. Also, I would like to thank my committee members Dr. Lawrence Wackett, Dr. Terrence Adam, and Dr. George Karypis for their very helpful insights, comments and suggestions at my proposal meetings. Additionally, I would like to acknowledge all of those people who provided technical support and assistance with running the web servers at the Minnesota Supercomputing Institute.

A special thanks to my parents, Yanqiu Hu and Qi Gao, my fiancée, Xiting Cao, and my friends for all their support during my time in graduate school.

Abstract

The University of Minnesota Pathway Prediction System (UM-PPS) (http://umbbd.msi.umn.edu/predict/) is a rule-based system that predicts plausible pathways for microbial degradation of organic compounds. Its biotransformation rules are based on reactions found in the University of Minnesota Biocatalysis/Biodegradation Database (UM-BBD, http://umbbd.msi.umn.edu/) or in the scientific literature. Since the UM-PPS was created in 2002, its rule base has grown to 275 entries. The original system predicted one level of prediction at a time. It provided a limited view of prediction results and heavily relied on manual interventions. It matched the query compound with all biotransformation rules one by one, which was a time-consuming process.

In 2008, the two-level visualization was first implemented to allow users to view two levels of predictions at a time. However, this visualization approach was usually not able to show the complete metabolism of a query compound, and users still needed expert knowledge to make educated choices to continue the prediction. In 2009, we started to develop a multi-level visualization and, simultaneously, work on increasing prediction speed. In 2010, the multi-level visualization was implemented to predict up to six levels of predictions at a time. Not only more products, but also common intermediates and cleavage products are displayed. Users can view prediction alternatives much more easily in a tree-like interactive graph. A multi-level prediction can be computationally intensive and requires users to wait longer than desired for the prediction results. Therefore, we used a multi-thread computing strategy that decreased the prediction run-time by half. We balanced the computing threads and pre-loaded all UM-PPS database tables to permit quick access to its data. Both of these improvements resulted in an additional 30% decrease in prediction run-time. We conducted a simulation study and used another web server to reduce the queuing interference by over 85%. Beta testers were satisfied with its visualization and performance.

The above improvements lead to a smarter and faster UM-PPS that has continued its growth in the past 4 years. It now displays better graphical results and predicts biodegradation pathway in a timely manner.

Table of Contents

List of Tables

List of Figures

Chapter I

INTRODUCTION

As xenobiotic compounds increasingly enter the environment, it is becoming imperative to understand their fate in soil and water. The environmental fate of chemicals is largely predicated on their biodegradation by microbes. Conducting biodegradation studies for all new chemicals would be prohibitively expensive. In this context, scientists and non scientists are increasingly relying on computational tools that predict biodegradation. Well-curated information on microbial biodegradation is the basis for developing knowledge-based systems for *in silico* predictions of metabolic pathways and accumulating end-products. This dissertation describes the improvements of a knowledge based biodegradation pathway prediction system.

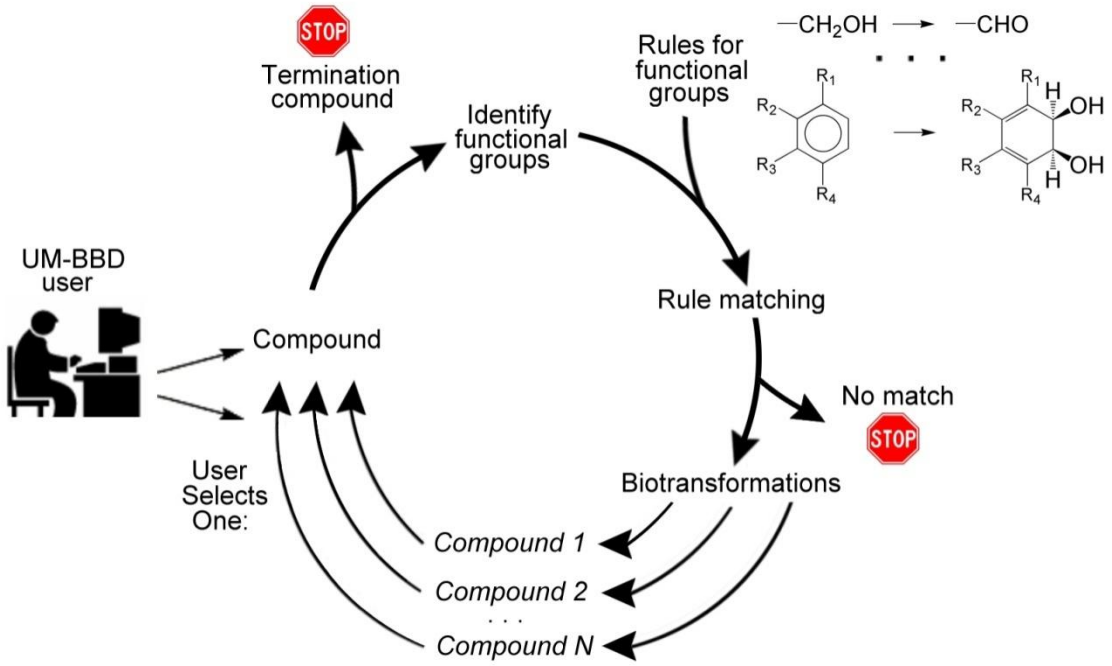
The last decade has seen increasing biodegradation studies reported in the scientific literature. The University of Minnesota Biocatalysis/Biodegradation Database (UM-BBD, http://umbbd.msi.umn.edu/) is a manually curated database containing information on microbial biocatalytic reactions and biodegradation pathways for primarily environmental xenobiotic, chemical compounds [1]. It has been freely available on the World Wide Web since 1995. As of December 2010, the UM-BBD had grown to about 200 biodegradation pathways along with over 1400 microbial catabolic reactions, over 1300 compounds, over 900 enzymes, and over 500 microorganisms.

One of the database's many features is that it allows the prediction of microbial catabolism of organic compounds using substructure searching and atom-to-atom mapping approach [2]. This feature is implemented in the University of Minnesota Pathway Prediction System (UM-PPS, http://umbbd.msi.umn.edu/predict/), a user-guided web-based application. Since its public release in 2002, the UM-PPS has served as an important biodegradation prediction tool. It is able to recognize organic functional groups in a compound and predict their transformations based on biotransformation rules, which are derived from reactions found in the UM-BBD database and scientific literature. We have reported content, methods, and the initial development of the UM-PPS in previous papers [2].

The UM-PPS is a rule-based system. Each biotransformation rule is assigned an aerobic likelihood. Aerobic Likelihood is the possibility that the reaction will occur under aerobic conditions, exposed to air, in soil (moderate moisture) or water, at neutral pH,

25°C, with no competing or other toxic compounds. All the UM-PPS biotransformation rules are assigned aerobic likelihood by two or more biodegradation experts in a five-point Likert scoring scale: Very Likely, Likely, Neutral, Unlikely, and Very Unlikely [3].

Figure 1-1. UM-PPS biodegradation pathway prediction cycle [4]**.**



The UM-PPS is a client-server type system that serves Internet users through its web interface. A user may submit a query compound by drawing it or directly entering a SMILES string [5]. If the compound is one of the pre-defined termination compounds, the prediction ends. Otherwise, the system will identify the functional groups in the query compound and match these groups against the system's rule-base. If no rule is matched, the prediction ends. If one or more rules are matched, the transformed products will be predicted. The user will then choose one of the products and the cycle continues, producing a multi-step biodegradation pathway (Figure 1-1). The UM-PPS can be accessed from the "Pathway Prediction System" link on the UM-BBD home page.

In 2002, Dr. Bo Kyeng Ho created the prototype system with 40 biotransformation rules. In December 2010, there were 275 biotransformation rules. As the number of rules increases, the UM-PPS predicts more products at each step. In a

multiple-step pathway prediction, this can easily cause combinatorial explosion, a problem that has been addressed previously [6]. To solve this problem, additional metabolic logic has been applied to remove implausible products in the predicted pathways. All 275 rules were assigned with absolute aerobic likelihood, 21 of them using the immediate feature, 123 with relative reasoning entries, 22 being super rules, and 27 with variable aerobic likelihood. The above four types of metabolic logic entities are described below.

Immediate feature. The system first introduced an immediate feature that guides users to most likely pathways as soon as possible by skipping one or more transient steps without manual intervention. This feature is only activated when the 'aerobic' option is chosen, and it is only applicable to all very likely and some likely biotransformations. If a user chooses to see all biotransformations regardless of the aerobic likelihood values, the immediate feature is turned off.

Relative reasoning. The UM-PPS uses relative reasoning as another effective approach to limit combinatorial explosion. To allow this feature, a relative reasoning field was added to the rule table, which can give certain rules priority over others. One relative reasoning entry decreased choices in prediction of aromatic ring degradation to 75% with no loss of sensitivity [2]. As the number of relative reasoning entries grows, the priority relationship among rules becomes complex. A function was developed to examine and report any potential futile loop in existing entries (e.g. Rule 1 has priority over Rule 2, Rule 2 over Rule 3 and Rule 3 over Rule 1).

Super rules. Besides the relative reasoning, the improved UM-PPS infrastructure allows super rules to provide additional reduction to the combinatorial explosion. A super rule was described as a combination of selected contiguous rules that form a small known metabolic pathway of its own. These rules can handle more than one biotransformation intermediates, guide users to some known pathways, such as β-oxidation, quickly and significantly shorten the prediction processes.

Variable aerobic likelihood. Since the UM-PPS uses a limited number of rules to make predictions of a wide range of different compounds, the aerobic likelihood of some rules is not always as accurate as expected. To make more plausible predictions, the system introduced variable aerobic likelihood values. To implement this feature, we

assigned regular expression patterns to selected rules. During a prediction process, structure-based aerobic likelihood values will be shown if a substrate triggers one or more rules and matches their patterns. This feature gives more accurate likelihood for rules triggered by substrates with certain chemical structures.

There are several systems similar to the UM-PPS. METEOR predicts mammalian detoxification metabolism [7]; CATALOGIC is a platform for models targeting environmental fate of chemicals [8]; and PathPred predicts enzyme-catalyzed metabolic pathways based on chemical similarity [9]. Among these systems, the UM-PPS is the only rule-based system that is freely accessible.

PROBLEM STATEMENTS

The original UM-PPS lacks an expandable visualization method. With more biotransformation rules added, the UM-PPS may predict more products at each step, and users have to choose from candidate products to continue the prediction. These manual interventions require expert knowledge to make educated choices. Since the original system only displays one predicted branch at a time, users cannot view or print the complete predicted pathway map. Duplicate products cannot be displayed concurrently since they are from different branches, and co-products predicted by cleavage biotransformations cannot be displayed together as a sub group. Overall, the original visualization method limited the understanding of biodegradation pathway prediction results.

With expanding UM-PPS rule-base, the computing efficiency for pathway prediction is slow. As the prediction step goes up, the time increases exponentially. If the system predicts a multi-level biodegradation pathway, such performance may be unacceptable for a web-based application. Thus, it is important to improve the prediction performance by taking advantage of the high-end web server.

GOALS AND OBJECTIVES OF THE DISSERTATION

The overall goal of this thesis is to improve the original UM-PPS on its prediction visualization and performance. The work includes solving problems inherent in the

original system, and adding new features to improve the system's functionality, accessibility and stability.

First, a new visualization method is developed that presents the complete biodegradation pathway in a directed acyclic graph (DAG). Second, a new computing method is designed to improve the system performance for multi-level pathway predictions.

The above objects aim to provide a better biodegradation prediction system that supports scientific research conducted by microbiologists, analytical chemists, and chemical risk assessors of environmental division of companies, academic institutes and government agencies. The improved system provides users a convenient tool to view and analyze the pathway prediction results. The pipeline system design might be applied in other metabolic pathway visualization or prediction applications.

GLOSSARY

The research work in this thesis includes several academic subject areas including microbial biology, expert system, statistics, graph visualization, and computing optimization. The following terms and definitions require clarification. They are in alphabetical order.

Aerobic likelihood: The possibility that the reaction will occur under aerobic conditions, exposed to air, in soil (moderate moisture) or water, at neutral pH, 25°C, with no competing or other toxic compounds.

Biodegradation pathway: A pathway consisting of consecutive enzyme-catalyzed reactions showing the breakdown intermediates starting from the substrate.

Biotransformation: The chemical modification made by a microorganism on an organic compound.

Biotransformation rule: A generalized reaction pattern describing one biotransformation or a group of similar biotransformations.

False positive (FP): The number of predicted products that cannot be found in the UM-BBD or known metabolism references.

False negative (FN): The number of products not predicted but observed in known metabolism references.

Intermediary metabolism: Intermediate steps in the chemical synthesis and breakdown by which foodstuffs is converted into cellular components.

Metabolic logic entity: A type of argument that specifies the attributes of biotransformation rules or the relationships between two or more biotransformation rules.

Microbial biocatalysis: The use of natural catalysts found in microorganisms to perform chemical transformations on organic compounds.

Microbial biodegradation: The breakdown process of organic compounds by the microorganism found in the environment.

Pathway prediction: The process of using biodegradation knowledge to predict plausible biodegradation pathway of an organic compound.

Prediction cycle: The procedure of matching a query compound with all biotransformation rules and returning transformed products.

Prediction step: A prediction step is the act of predicting biodegradation products after users click on the "Submit" or "Next" button. In one-step system, a prediction step contains one prediction cycle. In two-step system, a prediction step might contain multiple prediction cycles.

Queuing interference: Placing new submissions in a queue when the UM-PPS is occupied by a running prediction.

Relative reasoning: A higher priority that is assigned on certain biotransformation rules over others to exclude implausible biotransformations.

Sensitivity: The measure of the proportion of all actual products that are correctly predicted by the UM-PPS. The value is calculated by using TP/(TP+FN).

Specificity: The measure of the proportion of positive products that are predicted by the UM-PPS. The value is calculated by using TP/(TP+FP).

Super rules: A special rule that combines a group of consecutive rules that form a small pathway of their own.

Termination compounds: Primary products of a common intermediary metabolism, or dead-end compounds that accumulate in the environment.

Thread run-time: The time used for matching a query compound against the assigned biotransformation rule set.

True positive (TP): The number of predicted products that are observed in the UM-BBD or known metabolism references.

Variable aerobic likelihood: The changeable aerobic likelihood for a biotransformation rule that matches substrates with certain chemical structure features.

Xenobiotic compounds: Man-made chemicals that are foreign to an organism's normal biochemistry; they are present in the environment and pollute the environment in high concentrations.

DISSERTATION OVERVIEW

Chapter 2 – LITERATURE REVIEW – introduces the background of biodegradation pathway prediction; introduces graph theory; reviews current biological network visualization tools; reviews current biodegradation prediction tools and their visualization methods; discusses parallel computing, database caching, and queuing simulation approaches to improve the prediction performance.

Chapter 3 – TWO-LEVEL UM-PPS VISUALIZATION – describes one-level biodegradation pathway prediction systems; discusses the limitations of one-level visualization approach; describes a method used to produce a two-level prediction; discusses the limitations of two-level visualization approach.

Chapter 4 – MULTI-LEVEL UM-PPS VISUALIZATION – describes the complexity of metabolic pathway and the demands of better visualization; describes a new recursive method used to produce a multi-level prediction; presents an integrated pipeline system infrastructure and its application to the prediction of selected query compounds. The evaluation of the new visualization tool with a beta test survey is reported. Several possible improvements that might expand the system functionality are discussed.

Chapter 5 – IMPROVED UM-PPS PERFORMANCE – describes the current system usage and the demands of performance improvements; describes the hardware and software used to improve the prediction performance; presents the steps for extracting server usage statistics, stabilizing the server performance, implementing multi-

thread computing, and simulating queuing interference; reports the improved prediction performance of the multi-level UM-PPS.

Chapter 6 – CONCLUSIONS – The overall improvements of the UM-PPS in this dissertation is summarized; strengthens and weaknesses of the improved system are discussed; the future directions of the system development for biodegradation pathway predictions are outlined.

Chapter II

LITERATURE REVIEW

INTRODUCTION

Commercial chemicals are released into the environment with increasing amount, but only a small part of these chemicals have experimental data about their environmental biodegradability [10]. Since biodegradation experiments are usually prohibitively expensive, it is important to understand the environmental fate of these chemicals by using computational tools. In the past years, some computational methodologies have been explored, and a variety of biodegradation prediction tools have been developed. These tools assisted studies related to the environmental impact of chemicals undergoing biodegradation. However, improvements on biodegradation pathway visualization and computing performance are still in demanding by scientific researchers.

In this chapter, we will explore the topic of biodegradation pathway visualization, review a collection of software packages that can be used to improve the visualization, review selected biodegradation prediction tools and their visualization methods, and discuss computational frameworks that can be used to improve the prediction performance.

BIODEGRADATION PATHWAY VISUALIZATION

Visual graphs are more straightforward than word descriptions to understand biological data, but lack of proper software often poses a challenge for the visualization of biological data and its relationship. Indeed, scientific researchers often manually draw graphs to represent biological structures and processes. With the increasing amount and complexity of biological data, software visualization tools are in great need by researchers. Over the years, many visualization tools have been developed to interpret biological data. In this section, we will first briefly introduce graph theory and biological graphs, and then review a set of visualization tools.

Biological graphs

In graph theory, a set of nodes V and a set of edges E specify a graph G. Nodes may contain attributes including shape, size, and data sources, and edges may contain attributes including direction, type, and weights. Several special types of edges are useful to present relationship between nodes: a directed edge is an edge that specifies a head

node and a tail node; a multi-edge is a set of two or more edges that connect to the same end node; and adjacent edges are two edges that are connected by a node. Overall, a directed graph is a graph where all edges are directed edges, and a directed acyclic graph (DAG) is a graph without cycles.

Graphs can be used to describe biological entities and their relationships in biological systems. In a biological graph, the nodes are presenting basic biological entities; the edges are presenting the interactions or relationships between these entities. By using graphs, these interactions or relationships can be interpreted by using additional related measures, such as connectivity and distance.

The connectivity of a biological graph can be measured by using degree information. In a directed graph, in-degree describes the number of edges that reach an ending node, and out-degree describes the number of edges that go out from a starting node. For example, out-degree may present that a single gene encodes multiple protein isoforms. Meanwhile, the strength of a relationship between two nodes can be measured by using distance. The distance may indicate either the length of the shortest path between two nodes, or the number of paths between them. Based on distance matrix, standard analyses, such as clustering analysis, may be used to discover meaningful biological interactions.

Biological graphs can be used to represent biodegradation pathways. The biodegradation pathway visualization is concerned to describe all biodegradation intermediates, as well as with their relationships and to show how a compound is connected to another compound through a biotransformation rule. The complex interactions between these biological entities are often difficult to be described in words only, and the graph method provides a clear interpretation for such relationships. Additionally, different biological data types, such as compounds, reactions, enzymes, and other data are used in biodegradation pathways. In order to visualize such structure, the data types need to be linked and displayed in appropriate ways.

For example, nodes may represent a set of intermediates involved in a metabolic process, and edges may represent enzyme-catalyzed reactions modeling the relationships between all such intermediates. Directed edges may indicate the biotransformation directions, multi-edge may indicate common biodegradation intermediates, and adjacent

edges may indicate consecutive biotransformations. Overall, such a graph is able to provide a complete data structure to model the measured biological entities and relationships, and present meaningful biological knowledge.

If biological graphs are used in biodegradation pathway visualization, the graph-rendering components have to satisfy the following requirements. First, the graphs should be able to clearly present small structures and labels. Second, the graphs should be able to simplify the overall graph complexity and to provide an understandable layout of a pathway. Third, the graphs can be integrated in a pipeline system, where the graph-rendering component is used as a part of the workflow to produce graphical biodegradation results.

Current biological network visualization tools

In recent years, scientific researchers have developed a wide range of software packages that focus on visualizing biological network data based on 2D graphs to represent biological entities and their relationships. These include BioLayout Express [11], Cytoscape [12], GraphViz [13], Medusa [14], Ondex [15], Osprey [16], Pajek [17], Patika [18], PIVOT [19], and ProViz [20].

Several features are helpful to evaluate software for the visualization of complex biological networks. (1) application area that the software is suitable for; (2) release license that determines the usage and price; (3) visualization ability that determines graphical dimension, layout, and style; (4) programming interface that allows to be integrated with other systems; (5) cross-platform ability that specifies software compatibility; (6) documentation that trains developers; and (7) an active community that supports trouble-shooting and bug-fixing.

In this section, we will review current visualization tools that can be used to visualize the complex biological networks. Based on the above measures, we will discuss advantages, disadvantages, and relevance to biodegradation pathway visualization of these tools. These reviews will guide us to select the most appropriate tools for biodegradation pathway visualization.

A brief review of the 10 widely used, high-quality visualization tools listed above are summarized in Table 2-1. Some tools are more functional than others, and some

functions are more suitable for biodegradation pathway visualization. Preferred functions include the integration ability that allows other system to use graph rendering component through command-line interface (CLI) or application programming (API) interfaces, and layout ability that can produce DAG graphs. We will only discuss in more detail the following software, Cytoscape, Ondex, Pajek, and GraphViz, which satisfy the above requirements.

Table 2-1. Visualization tools feature review.

Software	License	Price	Application	Visualization ability					Cross-platform	Release			Active community	Documented
				Scale	Demension	Shape/Image	DAG layout	Hierarchical layout		GUI	CLI	APIs		
Cytoscape	LGPL	yFile library is not free	Molecular interaction networks	Large	2D	Both	Yes	Yes	Yes	Yes	Yes	Yes (Java)	Yes	Yes
Ondex	GPL	Free	Gene expression microarrays	Large	2D	Shape	Yes	No	Yes	Yes	Yes	Yes (Java)	No	Yes
Pajek	Own License	Free for academic users	Molecular interaction networks	Large	2D	Shape	Yes	Yes	Windows	Yes	Yes	No	Yes	Yes
GraphViz	CPL	Free	Telecommunication networks	Large	2D	Both	Yes	Yes	Yes	Yes	Yes	Yes (Java, unoffical)	Yes	Yes
BioLayout Express	GPL	Free	Microarray analysis	Large	2D/3D	Both	No	No	Yes	Yes	No	No	No	Yes
Medusa	GPL	Free	Protein-protein interaction	Medium	2D	Shape	No	No	Yes	Yes	No	No	No	Yes
Osprey	Own License	Free for academic users	Molecular interaction networks	Medium	2D	Shape	Yes	No	Yes	Yes	No	No	No	Yes
Patika	Own License	Free for academic users	Cellular process analysis	Large	2D	Both	Yes	No	Yes	Yes	No	No	No	No
PIVOT	Own License	Free for academic users	Protein-protein interaction	Large	2D	Shape	No	No	Yes	Yes	No	No	No	No
ProViz	GPL	Free	Protein-protein interaction	Large	2D/3D	Shape	No	Yes	Yes	Yes	No	No	No	Yes

GPL = General Public License; LGPL = Lesser General Public License; CPL = Common Public License; DAG = Directed Acyclic Graph; GUI = Graphical User Interface; CLI = Comand-Line Interface; and API = Application Programming Interface.

Cytoscape. It is a general-purpose modeling environment for integrating and visualizing biomolecular interaction networks, widely used for analyzing large data set of protein-protein, protein-DNA, and other genetic interactions [12]. The Cytoscape Core is developed under the GNU General Public License (GPL), and it is free for downloading and runs on all major platforms. This software supports hierarchical drawing of DAG graphs, and offers data manipulation features that allow users to view selected nodes and edges. It supports multiple widely used data formats, such as Systems Biology Markup Language (SBML) [21]. It runs on all major platforms and offers Java APIs that allow developers to build higher applications based on it. Complete documentations and active community are also available at its web site, thus developers can easily find trouble-shooting resources for software installation and development. However, Cytoscape renders hierarchical layout based on commercial software, yFiles Graph Library [22], which is not free for academic users.

Ondex. It is a tool for data integration and visualization, and it is free for academic users under GPL license. Ondex is designed to visualize transcription and protein interaction networks by extracting information from diverse biological data sets including biological databases and files. This software can render biological networks into directed graphs, and allow users to manipulate graphs by using filters to select specific data objects and relations. It accepts multiple input file formats including SMBL and free text, and outputs multiple graphics formats, such as BMP, EPS, JPEG, PDF, PNG and SVG. Ondex runs on all major platforms and provides Java APIs and documentations for application developers. However, it is not able to render graphs in hierarchical layout that presents level information in a directed graph. It also lacks an active community for users and developers on trouble-shooting and bug-fixing.

Pajek. It is a tool for visualization and analysis of large scale network. It can be used to visualize molecular, protein-receptor interaction, Internet, citation, or other general type networks. This software was developed by using Delphi (Pascal language) and runs on Windows platform. It is released under its own license that allows academic researchers to use it without charge. It provides a set of visualization tools that can represent the network by using different layout algorithms including hierarchical style. It

supports directed, undirected, or mixed graphs. It provides both GUI and CLI interfaces. Pajek also provides an R package that allows users to do statistical analysis. It uses mainstream input files, such as Microsoft Spreadsheets and free text files, and multiple graphical outputs formats. Since Pajek is not natively supported on a Linux system, it lacks cross-platform interoperability that allows to be used in other systems. An email list group is available to Pajek users and developers.

GraphViz. It is a collection of tools built on C/C++ libraries, and provides functionality for generating graph layouts and manipulating graph structures. It was first used to visualize telecommunication networks, but now it is used in general level visualization applications. GraphViz is developed by ATT developers under GPL license, and it is freely downloadable from its website. GraphViz can render either directed or undirected graphs in many layout styles including hierarchical and spring-out. It outputs multiple data formats, such as PNG, and SVG that can be embedded in interactive client side graphs and be presented in Hyper Text Markup Language (HTML) maps. GraphViz offers a graphical user interface (GUI) and a CLI interface. Based on the CLI, GraphViz can be integrated as a graph-rendering component in pipeline systems. This software is well documented and is supported by an active email list group, thus developers can easily find resources for trouble-shooting.

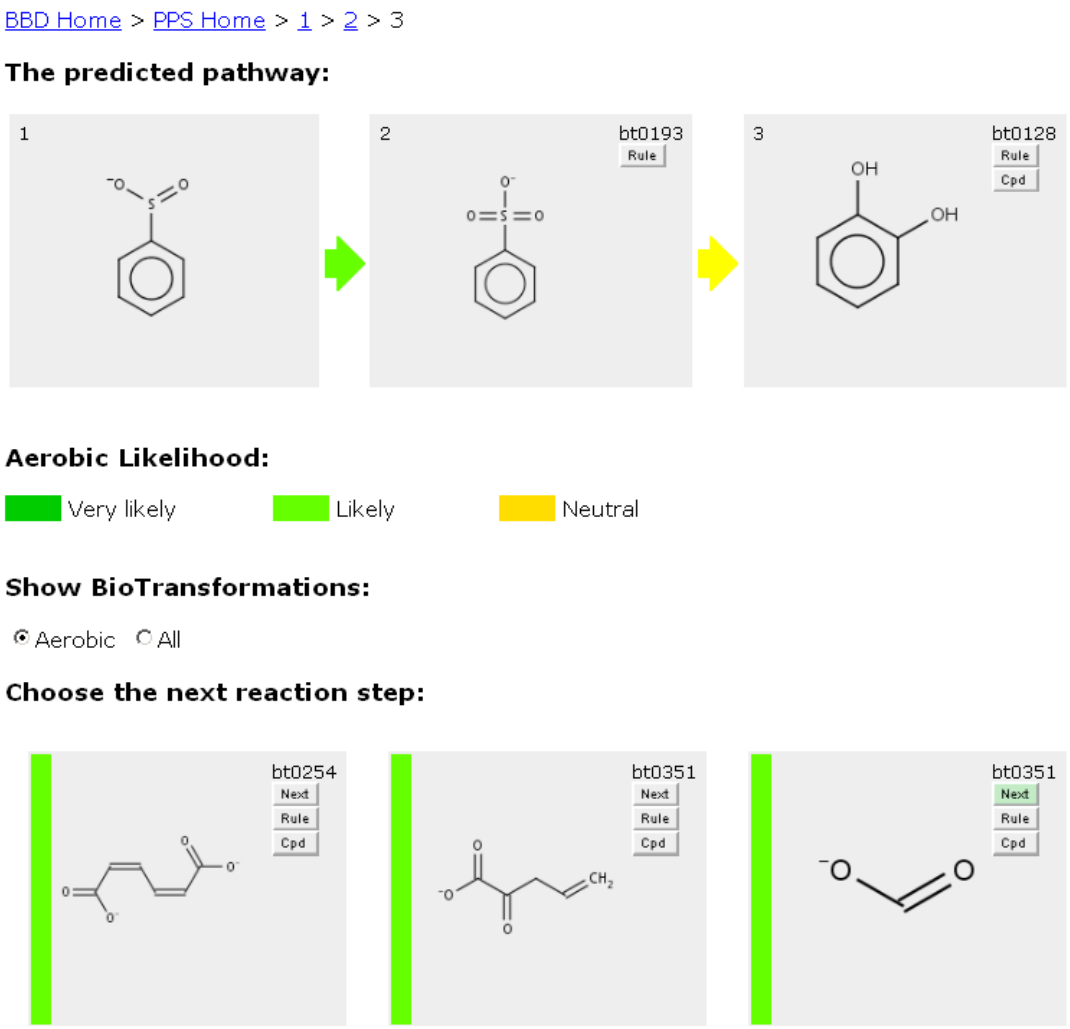
Based on the above review, GraphViz has all features needed to visualize biodegradation pathways. Thus, GraphViz was selected as the graph rendering component in this study.

Current prediction tools and visualization methods

Since the experimental data of many xenobiotic chemicals is not available, scientists usually predict environmental fate of these chemicals by using structurally similar chemicals whose metabolism is already known. Over the years, a wide range of predictions tools were developed to identify close structural analogs and enumerate plausible biodegradation products. In this section, we will review four selected tools, UM-PPS [2], Metabolizer [23], CATALOGIC [8], and PathPred [9], and their visualization methods.

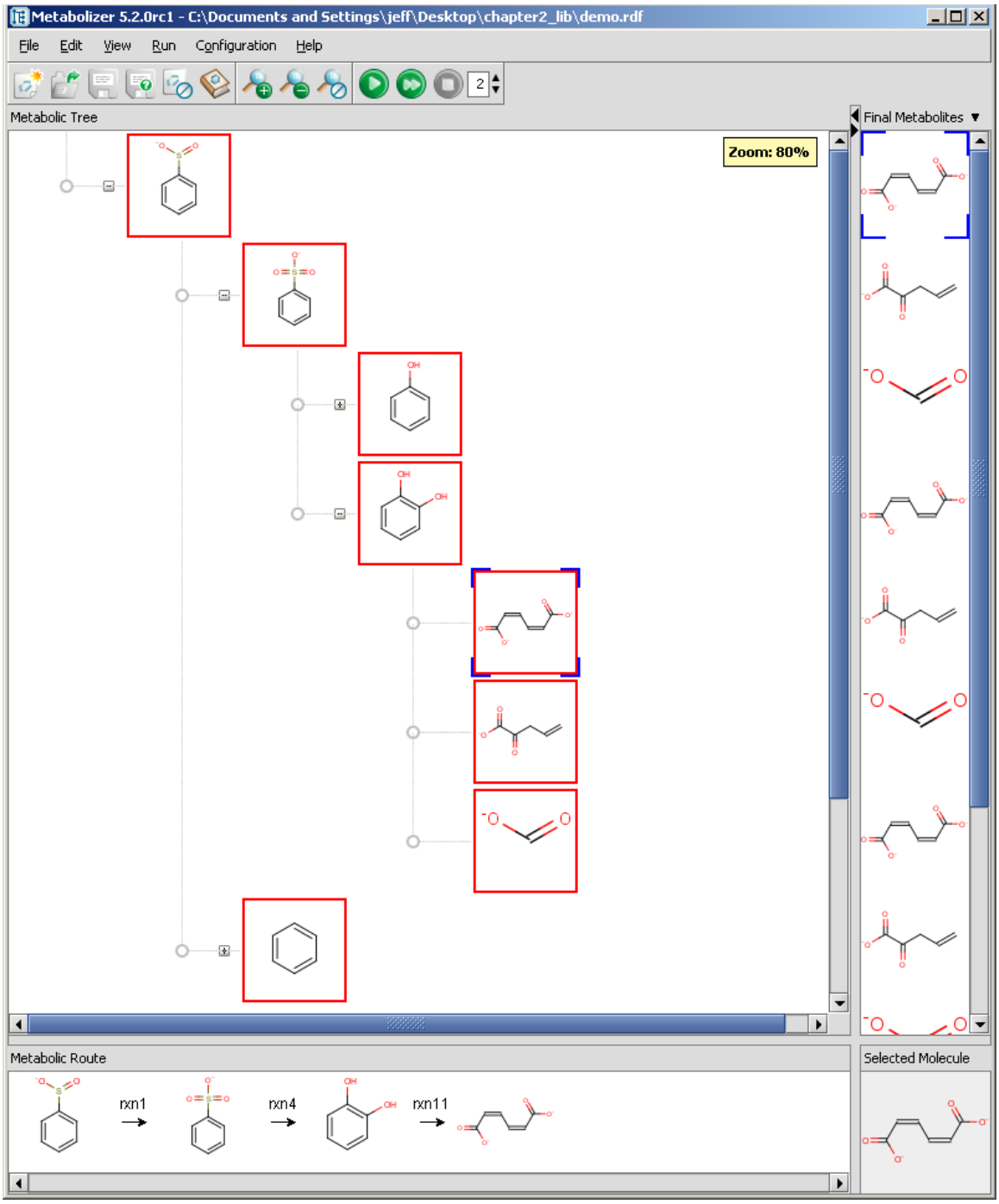
UM-PPS. The University of Minnesota Pathway Prediction System (UM-PPS) recognizes functional groups in chemicals that are potential targets of microbial catabolic reactions, and it predicts biotransformations of matched functional groups based on biotransformation rules [2]. It displays predicted pathway on its web interface, and provides links to external resources, such as, compounds, reactions, and biotransformation rules. The first version UM-PPS displays predicted products in a single biodegradation route. It lacks an expendable visualization interface that can display multiple biodegradation routes, and cannot clearly indicate common intermediates and cleavage biotransformations. The third step prediction of benzenesulfinate (C1=CC=C(C=C1)S(=O)[O-]), a compound not found in the UM-BBD, was displayed on the Pathway Prediction Results page (Figure 2-2).

Figure 2-2. The pathway prediction for benzenesulfinate in the UM-PPS.



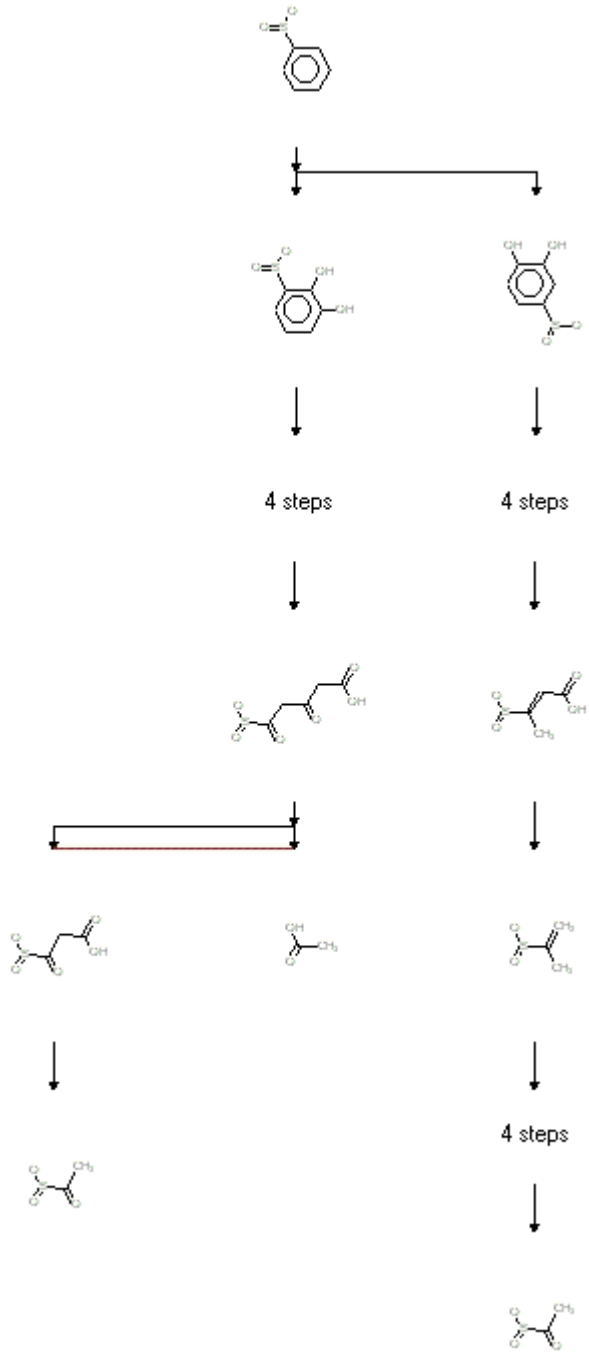
Metabolizer. This software predicts metabolic biotransformations by enumerating all the possible metabolites of a given compound, estimating metabolic stability, and displaying the major metabolites [23]. Metabolizer was developed by ChemAxon and built on the JChem Reactor engine. It has three release versions including a standalone application, CLI, and API libraries, and it is freely accessible to academic users. The software has a selective metabolism mode to only display dominating metabolites are produced by fast biotransformations and destroyed by slow ones. While the other exhaustive metabolism mode displays all metabolites and may results in a danger of combinatorial explosion. Users can specify the number of generations of metabolites to be predicted, and view the predicted biodegradation pathway in a hierarchical tree view. However, the Metabolizer doesn't have the capability to present common metabolites and cleavage biotransformations. The third step prediction of benzenesulfinate was displayed in the Metabolizer by using the UM-PPS biotransformation library (Figure 2-3).

Figure 2-3. The pathway prediction for benzenesulfinate in the Metabolizer.



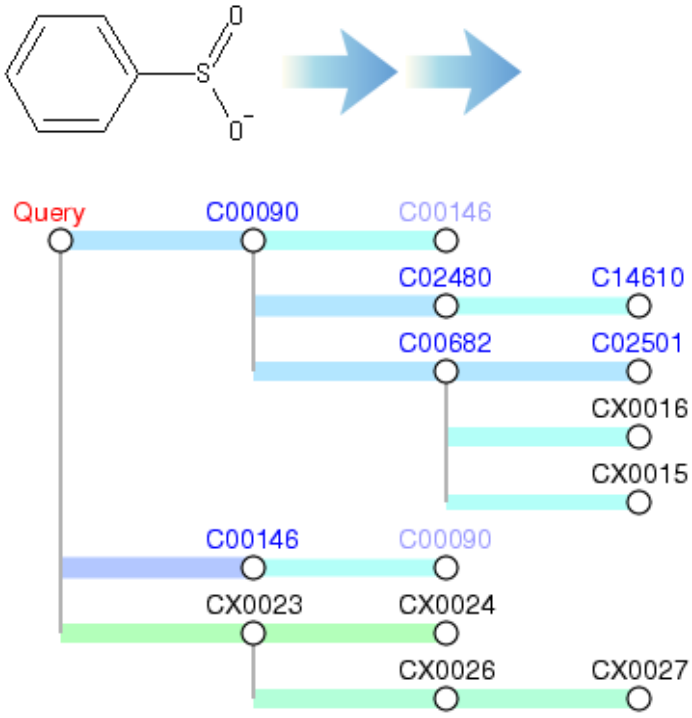
CATALOGIC. It is a platform for models targeting on environmental fate of chemicals. The software currently uses two categories of models, CATALOGIC models and CATABOL models, to predict plausible biodegradation within a specific time window, quantitative biological oxygen demand (BOD), and CO_2 production in different test conditions [8]. The software transforms the substrate to child products based on matched biotransformations and automatically repeats the cycle until a pre-defined probability threshold is reached. It displays the quantitative distribution of predicted metabolites in a hierarchical tree view. Cleavage biotransformations are marked with red bars. However, the software doesn't have the capability to remove duplicates and indicate common metabolites. The complete prediction of benzenesulfinate was displayed in the CATALOGIC by using the built-in Soil BioPath model (Figure 2-4).

Figure 2-4. The pathway prediction for benzenesulfinate in the CATALOGIC.



PathPred. The software was developed by KEGG developers and is freely accessible as a web-based tool. It uses a knowledge-based approach to predict enzyme-catalyzed biodegradation pathways of environmental compounds and biosynthesis pathways of secondary metabolites [9]. The software matches chemical analogs by using the SIMCOMP algorithm and predicts biotransformations based on the RDM patterns. It automatically repeats the prediction cycle until the maximal depth threshold is reached, and displays predicted pathways in a hierarchical tree view. But duplicates often disperse randomly in the pathway map. The common intermediates and cleavage biotransformations are not well indicated. The prediction pathway map and all paths of benzenesulfinate were displayed in Figure 2-5.

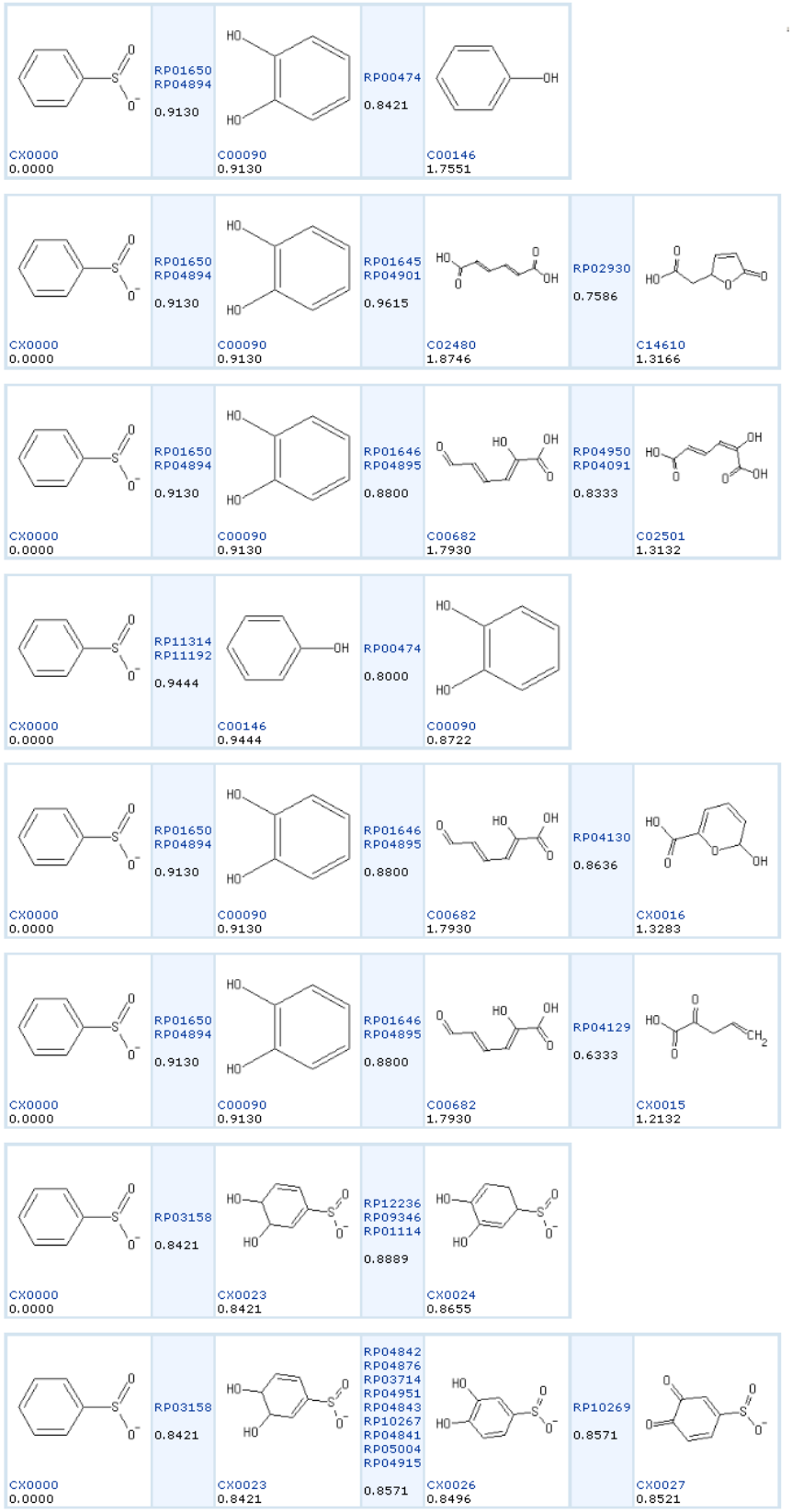
Figure 2-5. (a) The predicted pathway map for benzenesulfinate in the PathPred.



Path List <show all path>

2 <show path> CX0000 (R) C00090 (R) C00146
3 <show path> CX0000 (R) C00090 (R) C02480 (R) C14610
3 <show path> CX0000 (R) C00090 (R) C00682 (R) C02501
2 <show path> CX0000 (R) C00146 (R) C00090
3 <show path> CX0000 (R) C00090 (R) C00682 (R) CX0016
3 <show path> CX0000 (R) C00090 (R) C00682 (R) CX0015
2 <show path> CX0000 (R) CX0023 (R) CX0024
3 <show path> CX0000 (R) CX0023 (R) CX0026 (R) CX0027

Figure 2-5. (b) The predicted paths for benzenesulfinate in the PathPred.



The above four biodegradation prediction tools have their own strengths but they have some common weaknesses on their visualization methods. First, the duplicates are not properly handled. The UM-PPS simply removes duplicates from its prediction results; the Metabolizer and CATALOGIC displays all duplicates and their child pathways; and the PathPred terminates further pathways for duplicates, but different duplicates are not marked by unique identifiers. Second, common metabolites are not clearly indicated by all four prediction tools. The UM-PPS only displays one level predict at a time and displays common metabolites on different results pages; the Metabolizer displays common metabolites as duplicates in its metabolites list; the CATALOGIC only displays quantitative description of all metabolites; and the PathPred displays common metabolites randomly in the predicted pathway map. These important features are demanded in the improved UM-PPS visualization.

PREDICTION PERFORMANCE

Parallel computing. The UM-PPS uses the ChemAxon JChem suite as its core component [24]. The JChem Reactor is used as the virtual reaction engine to predict biotransformations by matching the substrate against UM-PPS rules. It runs one biotransformation rule at a time, and runs as many times as the total number of rules in a sequence. When the rules increased, this serial computing strategy became a performance bottleneck.

Parallel programming grew out of the dramatic technology progress in the computer hardware and software. Today multi-core or multi-processor systems are inexpensive and plentiful. Many parallel computing frameworks are implemented in different programming languages. But no reviewed biodegradation prediction tool supports parallel computing by taking the advantage of the available resources.

Message Passing Interface (MPI) is an industrial standard for parallel computing on computer clusters. It enables communication between multiple processors through specific commands [25], and allows developers to build more complex computing infrastructures. Many MPI tools are implemented in FORTRAN, C/C++, or other languages. Although Java does not have an official MPI binding, there have been several attempts to bridge Java and MPI, with different degrees of success and compatibility.

These attempts include Bryan Carpenter's mpiJava [26], MPJ API [27], and P2P-MPI [28].

Java language also provides its own synchronization mechanism to support multi-thread programming. It was implemented in a Java Thread Class and an individual package, Java Concurrent Classes [29], which are able to split a program into multiple simultaneously running tasks. This framework is suitable for a single processor with multiple cores, or multiple processors that run their own threads simultaneously [30].

The above frameworks provide many ways to build parallel computing infrastructure for the UM-PPS. However, MPI approach and multi-thread approach have their own set of strengths and weaknesses. MPI approach is usually used for large data set, but more effort is needed to put into the development and maintenance. While Java multi-thread approach can be easily developed and deployed to small or medium projects. Its simplicity provides a better understanding of process synchronization, such as resource sharing and deadlock detecting. Since the UM-PPS does not handle large data, and the multi-core UM-PPS server is ready for multi-thread implementation, the Java multi-thread approach is selected in this study.

Caching database. When a web-based application frequently queries its database, it is important to apply caching strategy to improve the performance. Caching makes database more efficient, as it can deliver the most frequently requested data from a high-speed, in-memory cache, and the application will not perform the query operation twice. Many bioinformatics databases, such as WormBase [31], CONDOR [32], and Algal [33], have successfully used caching strategy to speed up the data loading process. Different systems may use different caching strategies for improving the performance. In Tomcat server, web session objects can be used to store database tables and create dynamic web cache. Such implemented database caching significantly increased the stability of web server and improved web server throughput [34]. The UM-PPS uses MySQL 5.0 relational database and Tomcat 5.0 web server that provides web session objects for caching frequently used database tables.

Queuing interference. Besides parallel computing, it is also important to evaluate the system usage and reduce the overuse. Since more than one Internet user may access a web-based system at any time, competitive requests often cause overuse on the limited

computing resource. To address this problem, server side log file can be used to determine the frequency of system usage, statistical tools can be used to measure the frequency by modeling the prediction run-time distribution, simulation tools can be used to simulate the prediction processes and evaluate the potential overuse, and hardware resources can be allocated to serve the additional requests. Using additional servers to successfully handle frequently accessed pages, such as BLAST request and query pages, has been previous reported [31].

CONCLUSION

Given the growing amount of biological data, the need of better visualization and computing performance will only increase in the foreseeable future. The current biological data visualization software provides a wide range of ways to facilitate biodegradation prediction results presentation. The current biodegradation tools used various visualization methods and provided different features that can be improved and integrated for a better prediction. Multi-thread computing, database caching, and queuing simulation approaches are also available to improve the biodegradation prediction performance.

Chapter III

TWO-LEVEL UM-PPS VISUALIZATION

Note, the material in this chapter has been adapted and expanded from our publication in Nucleic Acids Research [1].

INTRODUCTION

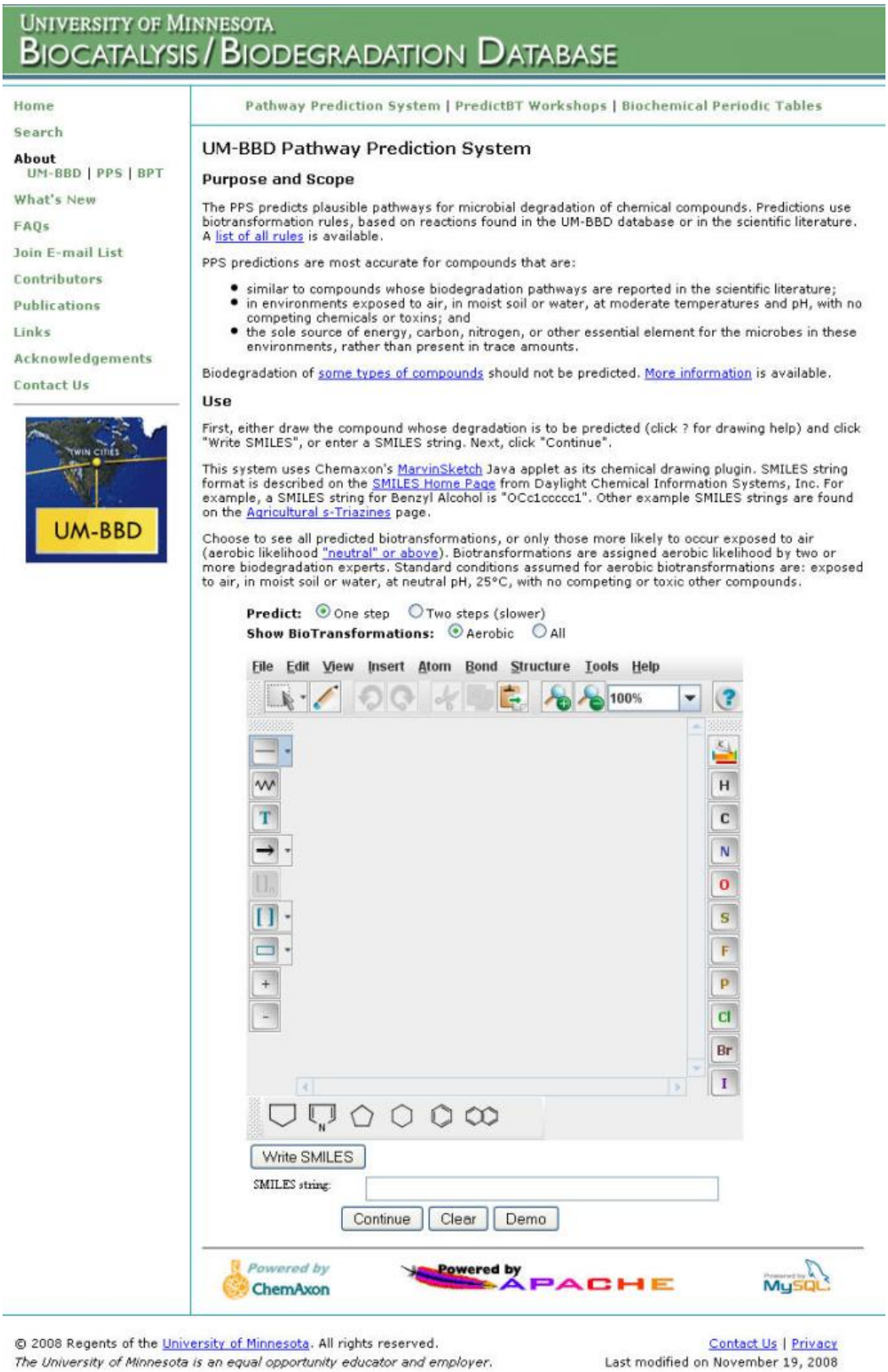
The goal of UM-PPS visualization is to visualize predicted biodegradation pathways consisting of metabolites, biotransformations, and their relationships in a proper way, so users can easily view and analyze the pathway prediction results. In fall 2002, a prototype of the UM-PPS was initially developed, which used one-level visualization approach to display prediction results for only one generation.With the increased demand for better visualization and an upgraded hardware configuration, we developed two-level visualization in 2008. In this Chapter, we will introduce the one-level UM-PPS visualization approach and its limitations, and describe the two-level visualization approach with its strengths and weaknesses.

One-level UM-PPS Visualization

This visualization approach was initially developed to display one level pathway prediction results for a chosen compound. Users accessed this visualization tool from the UM-PPS home page by using a web browser which supports Java [35] and JavaScript [36]. They could draw chemical structures using ChemAxon MarvinView [24] tool and generate SMILES strings by clicking on "Write SMILES" button or enter SMILES strings directly [5]. They could also click on the "Continue" button to submit a compound and start a prediction. The UM-PPS home page and its compound input area are depicted in Figure 3-1 (next page).

Before started a prediction, the system would validate the SMILES string and exclude submissions with the following errors: an empty SMILES string; a SMILES string of multiple compounds or salt; a SMILES string in incorrect format; a SMILES string of a chemical reaction; a SMILES string of polymers or chemicals with a molecular weight greater than 1,000; a SMILES string that contains one or more aromaticity errors; a SMILES string that contains atoms rather than C, H, N, O, P, S, F, Cl, Br, and/or I atoms; a SMILES string that does not contain a C atom; and a SMILES string that contains valence errors.

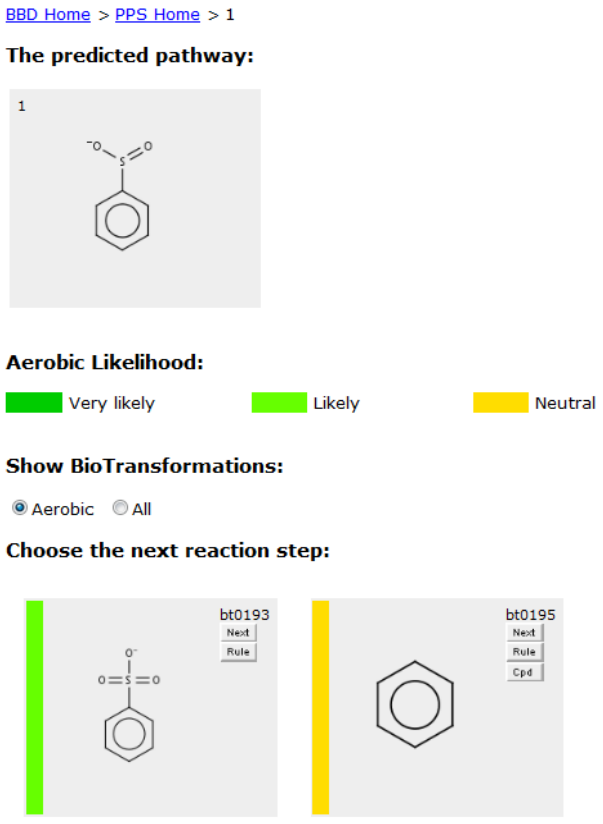
Figure 3-1. The UM-PPS home page



When a compound passed the validation by the UM-PPS, its organic functional groups were recognized and might trigger one or more biotransformation rules, which predict one or more reactions. This process took a few seconds to complete. When it was done, the prediction results were immediately displayed on a Pathway Prediction Results page for users to review. In Figure 3-2, we illustrated one-level visualization interface by using the prediction results of benzenesulfinate (C1=CC=C(C=C1)S(=O)[O-]), a compound not found in the UM-BBD.

Figure 3-2. The Pathway Prediction Results page. Three web pages show (a) the first step prediction, (b) one of the third step predictions, and (c) another choice of the third step predictions for benzenesulfinate.

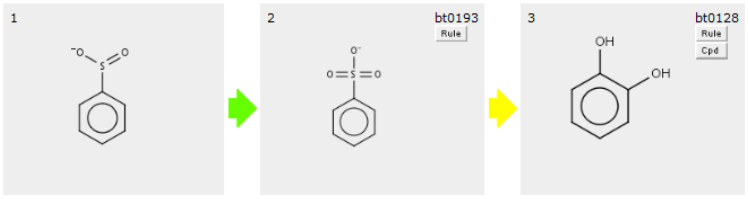
(a) The first prediction step for benzensulfinate



(b) One of the third prediction step for benzensulfinate

BBD Home > PPS Home > 1 > 2 > 3

The predicted pathway:



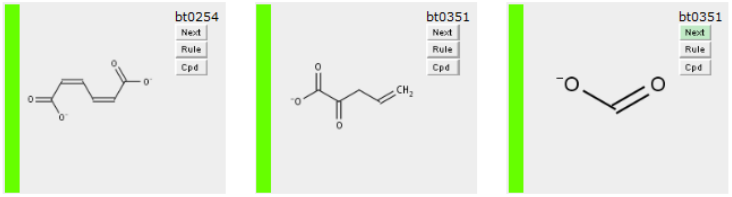
Aerobic Likelihood:

Very likely Likely Neutral

Show BioTransformations:

Aerobic All

Choose the next reaction step:



(c) Another choice of the third prediction step for benzensulfinate

BBD Home > PPS Home > 1 > 2 > 3

The predicted pathway:

Aerobic Likelihood:

Very likely Likely Neutral

Show BioTransformations:

Aerobic All

Choose the next reaction step:

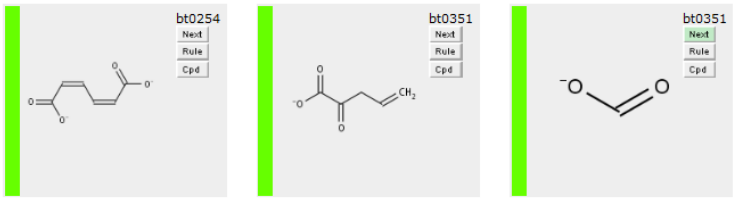


Figure 3-2a shows the first step prediction for benzenesulfinate in a Pathway Prediction Results page. The substrate, benzenesulfinate, triggered two biotransformation rules, bt0193 and bt0195, which transformed the substrate to two products, benzenesulfonic acid and benzene, respectively. Users may choose either product and continue the prediction. Figure 3-2b and Figure 3-2c showed two different choices of the third step prediction for benzenesulfinate. The selected pathway is displayed on the top of the web page. The predicted products of the current step are displayed below the pathway, and ordered by aerobic likelihood followed by biotransformation rule ID, from small to large. The aerobic likelihood of a triggered biotransformation rule is shown in a colored arrow or a vertical bar for selected pathways or predicted products respectively. The color of aerobic likelihood is explained in the "Aerobic likelihood" legend. The one-level visualization handles two special types of product, duplicate products and cleavage products, and includes some user-friendly interface features.

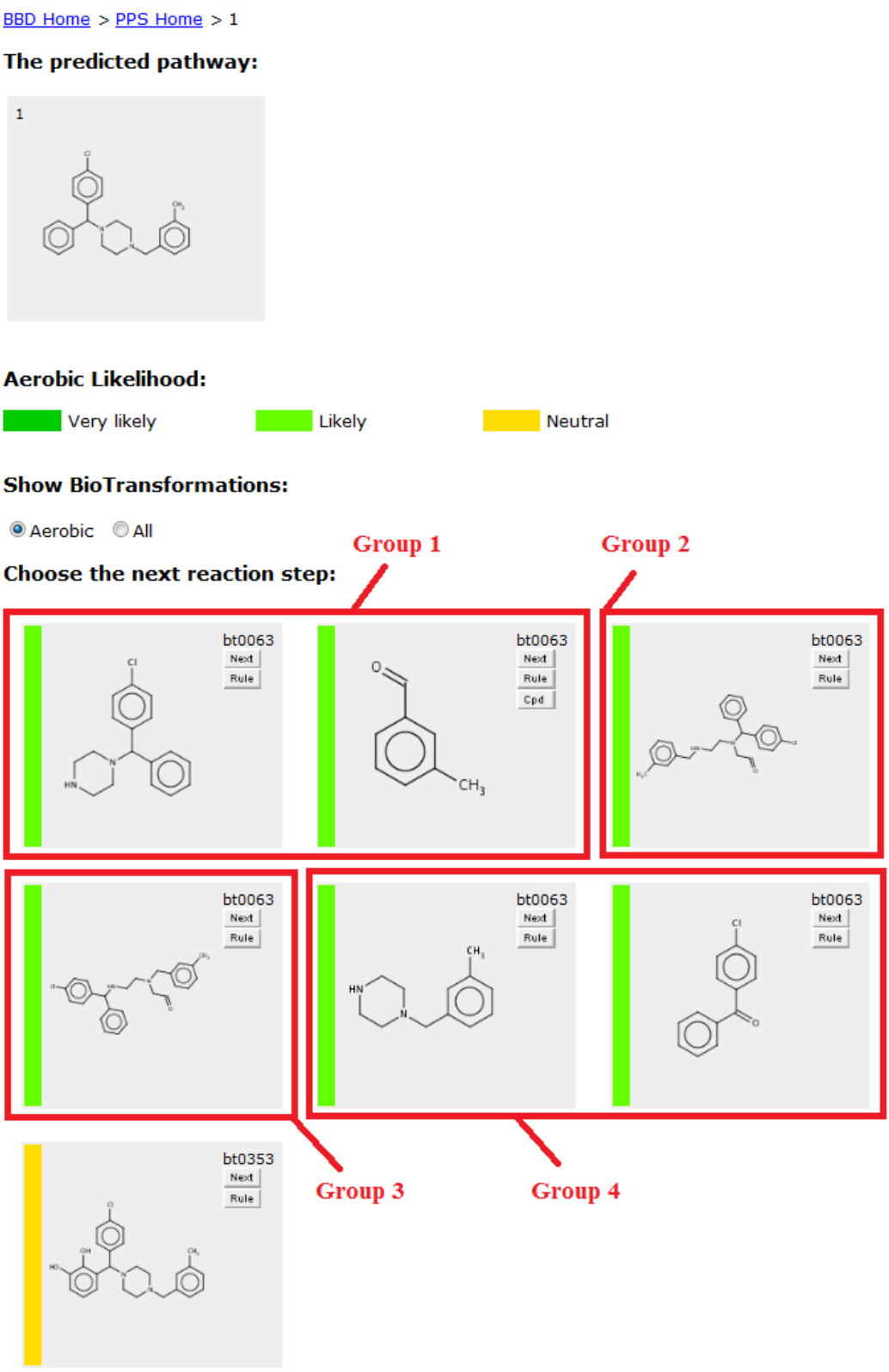
Duplicate products. The one-level visualization approach does not show duplicates in a Pathway Prediction Results page. But users can view duplicates on different pages of selected prediction results. For example, from the first step prediction results set of benzenzesulfinate (Figure 3-2a), users choose either benzenesulfonic acid or benzene to continue the prediction, catechol would always be predicted by either of two biotransformation rules, bt0128 (Figure 3-2b) and bt0005 (Figure 3-2c), respectively.

Cleavage products. Cleavage products are predicted by biotransformation rules that transform one substrate to two smaller products. In one-level visualization system, cleavage products are assigned with the same biotransformation rule IDs and displayed in adjacent positions. For example, Figure 3-2b and Figure 3-2c showed two alternatives of the third step predictions for benzenesulfinate and choices for the fourth step. In both alternatives, catechol (Compound 3 in Fig. 3-2b and Fig. 3-2c) triggered two biotransformation rules, bt0254 and bt0351, that predicted three products. One of them, bt0351, is a cleavage rule that transformed catechol to two co-products, 2-oxopent-4-enoate and formate, which were displayed together at the end of the choices.

Another example was seen in the prediction of a compound (Cc1cccc(CN2CCN(CC2)C(c2ccccc2)c2ccc(Cl)cc2)c1) as depicted in Figure 3-3. This substrate triggered two biotransformation rules, bt0063 and bt0353, at the first prediction

step. One of them, bt0063, predicted four reactions that transformed the substrate to six products. Two of these four reactions are cleavage reactions that transformed one substrate to two products. Thus the products can be manually categorized into four groups as shown in Figure 3-3. However, cleavage products were not indicated by reaction information but displayed with other transformed products.

Figure 3-3. The Pathway Prediction Results page for cleavage products.



Interface features. We improved the original system interface by adding some features on the page layout, navigation bar, and graphical component. In the upper right corner, the “Rule” button which is linked to the specific UM-BBD biotransformation rule page was added. The “Cpd” (Compound) button was added when any predicted product has a case sensitive match of SMILES string with a UM-BBD compound. These changes show connection between the UM-PPS and the UM-BBD. The green “Cpd” was also added when any predicted product can be found in the KEGG PATHWAYs.

Since the users need to view previous prediction results, we added a navigation bar (for example, "BBD Home > PPS Home > 1 > 2 > 3") on the top of the Pathway Prediction Results page depicted in Figure 3-2. This bar permits quick navigation between different prediction steps and the UM-PPS and UM-BBD home pages. The numbers after the “PPS Home” link stand for the prediction steps. If the users want to continue the pathway, they may click the "Next" button on any predicted alternative in the results list. The links on step numbers allow users to go back to previous prediction results pages easily.

The original one-level system displayed predicted compound by using Java Applets [37]. Many Java Applets often increase the web page loading time or even crash web browsers. In 2007, we improved the one-level visualization by replacing the Java Applets with PNG [38] graphics. The improved interface is more stable and compatible with all major web browsers, while the Java Applets are still available in pop-up windows when users click on the static compound graphics.

One-level UM-PPS limitations

The one-step system displays only a limited number of products for one level of biodegradation prediction and a user has to manually choose a product to continue the prediction. The system does not show duplicates on a Pathway Prediction Results page. Cleavage products are not well indicated, but displayed among other transformed products.

METHODS AND MATERIALS

In May 2007, a UM-PPS user asked that we represent predictions as a metabolic tree. It was not possible to do this rapidly enough for interactive use at that time. With the upgrade of hardware configuration, we decided to improve the system to allow users to view a two-level prediction as a compromise solution.

Hardware and Software

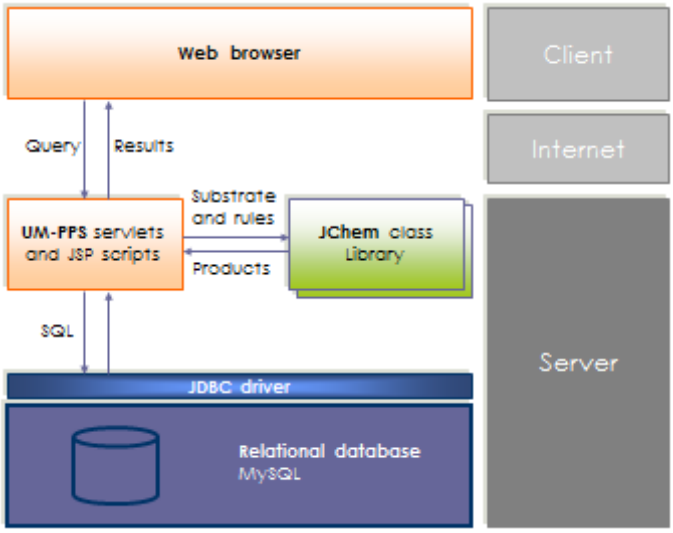
Since 2007, the UM-BBD and UM-PPS have been hosted at a high-performance server with a Linux 32 bits operating system in the SDML (Scientific Data Management Laboratory) of the Minnesota Supercomputing Institute (http://www.msi.umn.edu/). The UM-PPS server includes a database server, MySQL version 5.0.26 [39], Java development and run-time software [35], a set of web server applications, Perl [40], Apache Web Server [41], and Apache Tomcat Server version 5.5.28 [42].

The application layer of the UM-PPS is coded in JSP (Java Server Pages) [43] and Java servlets [44]. Core functions are coded in Java and are built upon the ChemAxon JChem software 5.0 [24]. Functions are called by the application layer which controls the whole pathway prediction process. The web-based graphical user interface of the UM-PPS is implemented by the ChemAxon Marvin Sketch Java applet, ChemAxon Marvin View Java applet, and ChemAxon Molecule Class.

RESULTS

In August 2008, we implemented the two-level visualization approach to allow users to view a two-level prediction at a time. As depicted in the new system infrastructure (Figure 3-4), the control Java program will automatically run predictions by using JChem class library for two cycles before it returns results to web browsers. Thus, the two-level visualization approach enables better view and analysis of plausible choices.

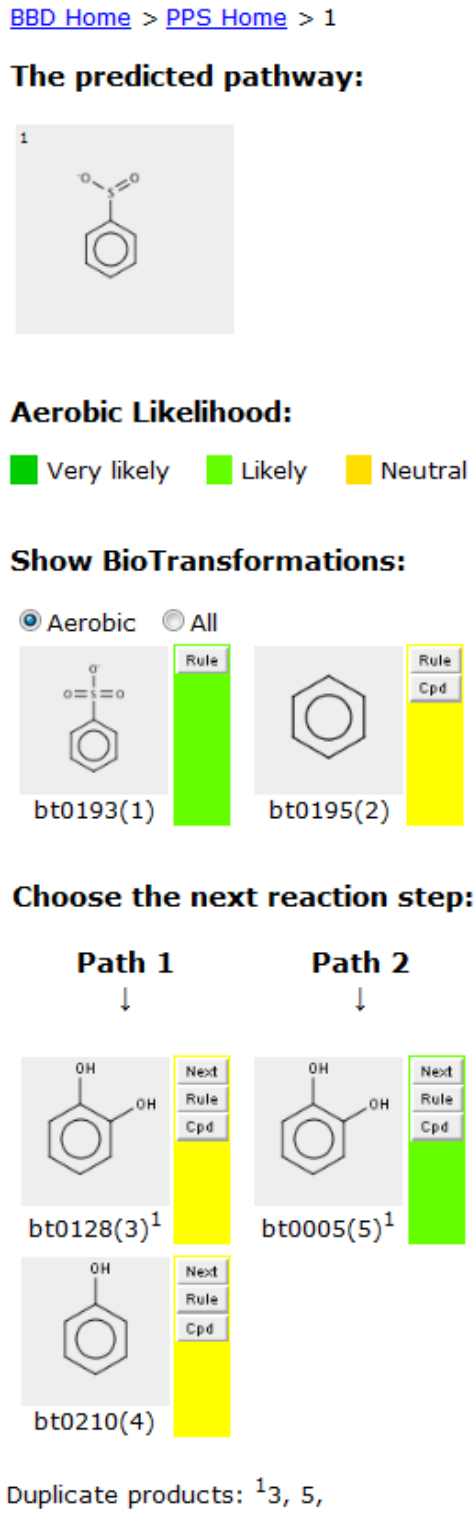
Figure 3-4. Two-level pipeline system infrastructure.



For example, Figure 3-5a showed a two-level Pathway Prediction Results page for benzenesulfinate. In the first prediction level, the Pathway Prediction Results page showed two products, benzenesulfonic acid (Compound 1 in Fig. 3-5a) and benzene (Compound 2 in Fig. 3-5a). In the second prediction level, the Pathway Prediction Results page showed two following paths that started from benzenesulfinate acid and benzene respectively. In Path 1, benzenesulfonic acid went through two neutral biotransformations to catechol (compound 3 in Fig. 3-5a) and phenol (compound 4 in Fig. 3-5a), and in Path 2, benzene went through a neutral transformation to catechol. The selected pathway is displayed on the top of the web page as in the one-level visualization interface. The products and following paths of the current prediction step are displayed below the selected pathway, and ordered by aerobic likelihood followed by biotransformation rule IDs, from small to large. The aerobic likelihood of triggered biotransformation rules and the "Aerobic likelihood" legend are presented below the selected pathway as well. The improved two-level system can better present two special products, duplicate products and cleavage products, and includes more user-friendly interface features.

Figure 3-5. The two-level Pathway Prediction Results page. Two web pages show (a) the first step prediction and (b) the third step prediction of benzenesulfinate.

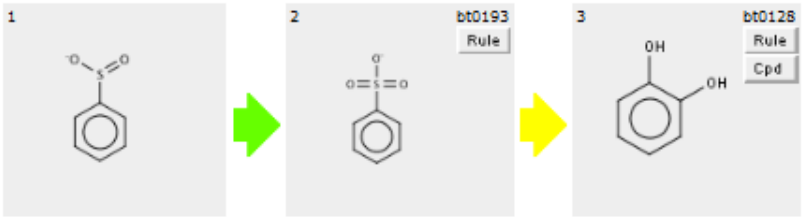
(a) The first step prediction of benzenesulfinate



(b) The third step prediction of benzenesulfinate

BBD Home > PPS Home > 1 > 2 > 3

The predicted pathway:

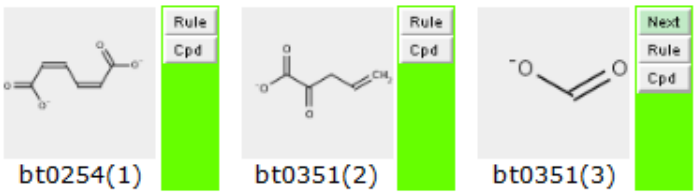


Aerobic Likelihood:

Very likely Likely Neutral

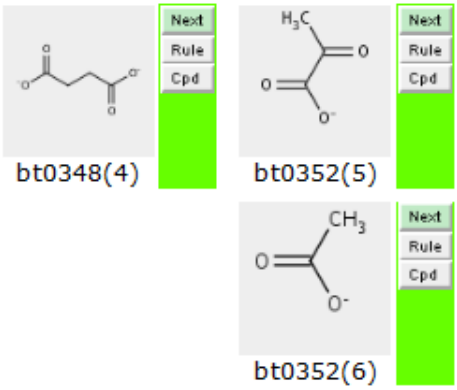
Show BioTransformations:

Aerobic All



Choose the next reaction step:

Path 1 **Path 2**



Duplicate compounds. The two-level visualization approach shows duplicates as common intermediates on the second level of prediction. Since this approach can display two levels of predictions at a time, users can easily view duplicates located on different paths. For example, benzenesulfinate acid and benzene started two paths in the second level of prediction as depicted in Figure 3-5a. Both paths contained the product, catechol, which was labeled as Compound 3 and Compound 5 in Path1 and Path2 respectively. These two duplicate products were indicated in the footer note at the bottom of the Pathway Prediction Results page, thus users can easily found Path 1 merged with Path 2. This analysis showed that both paths transformed the substrate benzenesulfinate to the common product catechol. Path 2 was the shortest pathway and might be the best choice to continue the prediction.

Cleavage products. As describe previous, cleavage products are predicted by cleavage biotransformation rules that transform one substrate to two smaller products. In two-level visualization system, cleavage products are assigned with the same biotransformation rule IDs and displayed in adjacent positions in a path. For example, Figure 3-5b showed two biotransformations of catechol, bt0254 and bt0351, and three predicted products at its first level of prediction. One of them, bt0351, is a cleavage rule that transformed catechol to two co-products, 2-oxopent-4-enoate and formate, which were displayed in adjacent positions at the end of the choices. One of them, 2-oxopent-4-enoate, also triggered another cleavage rule, bt0352, which predicted two smaller products, pyruvate and acetate, in Path 2.

Interface features. The two-level system home page is similar to the one-level system described previously [1]. Users had an option to use two-level visualization tool by clicking on the "two-steps" check box. The two-level visualization tool displays the first level products as intermediates and enumerates the following level products in each path. The tool displays ordered compound IDs for compounds in both selected pathways and predicted choices. The interface shows duplicate products by using incremental superscripts and a footer note. Since the system predicts two levels of prediction at a time, the navigation bar will forward two steps after a prediction is complete. Users are also allowed to go back one step to view intermediate results.

Two-level UM-PPS limitations

Since common metabolites are randomly displayed in a Pathway Prediction Results page, there is not a straight forward way for users to identify them. Although all such products are indicated in the "duplicate products" footer note, users still have to make extra effort to find them by using compound IDs and duplicate group IDs. Cleavage products are still shown among other products, thus users usually have difficulties to view them without special labels. The two-level visualization approach is not implemented in a hierarchical structure, and it is not expendable to visualize larger prediction results that contain more than two levels of predictions.

CONCLUSION

The one-level visualization approach only displays a limited number of predicted products. The two-level visualization approach can display more products by predicting two consecutive reactions at a time, but this approach does not usually show complete metabolism of the substrate. Both visualization approaches require users to make extra effort to identify common metabolites and cleavage products. The prediction process also requires users to have extensive knowledge of biodegradation and make educated choices as to the best pathway to pursue further.

Chapter IV

MULTI-LEVEL UM-PPS VISUALIZATION

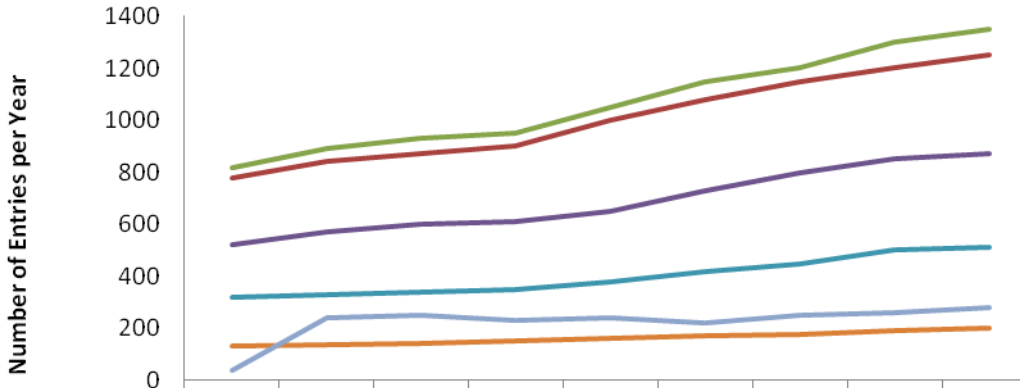
The material in this chapter has been adapted and expanded from our publication in Nucleic Acids Research (NAR Manuscript 2011).

INTRODUCTION

Complexity of Metabolic Pathway Demands Visualization

The information in the UM-BBD has been in linear growth over the last eight years. Compounds and reactions have been growing fastest among all types of data. Enzymes and microorganisms have been growing slower than compounds and reactions, since one enzyme or microorganism may be associated with more than one compound or reaction. Pathways have been growing slowest, since a pathway is the biggest data type that contains many child data types including compounds, reactions, enzymes, and microorganisms. Overall, the average annual growth rate is 9.5%. All statistics are shown in Figure 4-1.

Figure 4-1. The UM-BBD and UM-PPS growth from 2002.



	2002	2003	2004	2005	2006	2007	2008	2009	2010
Compound	780	840	870	900	1000	1080	1150	1200	1250
Reaction	820	890	930	950	1050	1150	1200	1300	1350
Enzyme	520	570	600	610	650	730	800	850	870
Microoganism	320	330	340	350	380	420	450	500	510
Pathway	130	136	140	150	160	170	178	190	200
Btrule	40	240	250	230	240	220	250	260	280

Although the growth of UM-PPS biotransformation rules has been fairly slow, an approximately 20% overall growth was seen in the metabolic logic entities since its last report in September, 2009 [1]. In December 2010, there were 275 total biotransformation rules with absolute aerobic likelihood, 21 of them using the immediate feature, 123 with relative reasoning entries, 22 being super rules, and 27 with variable aerobic likelihood. The increasing complexity of predicted metabolic pathway demands a better visualization approach.

In 2007, with an update of our server, we were able to show two prediction levels in a timely manner [1]. In 2009, we started to develop a multi-level prediction system and, simultaneously, work on increasing prediction speed. This system was made public in August 2010. It allows users to visualize all plausible products and pathways in a directed acyclic graph (DAG).

METHODS AND MATERIALS

Hardware and Software

Since 2010, the UM-PPS have been hosted at two sets of high-performance servers in the Minnesota Supercomputing Institute. The programming languages, database, and web server configurations were described in Chapter 3.

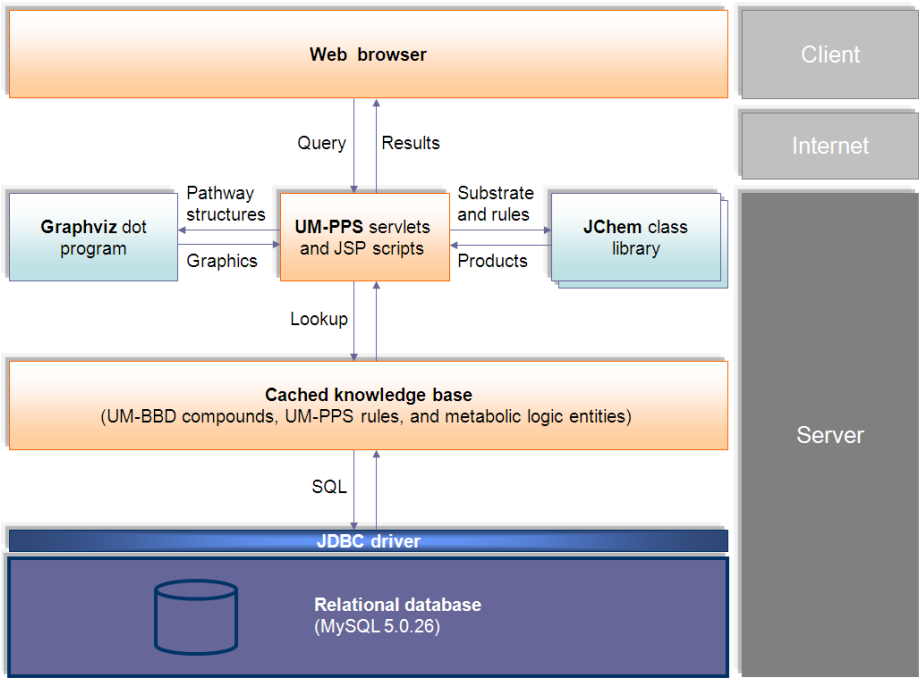
The UM-PPS is coded in JSP [43] and Java servlets [44]. We updated the Chemaxon JChem software to version 5.3.5 [24]. We use Graphviz software 2.27 [13] as the graph rendering engine to generate graphical pathways maps based the predicted pathway structures. We use Tomcat application scope as a database cache layer to store UM-BBD compounds, UM-PPS rules, and metabolic logic entities. AJAX [45] is also used to enhance the graphical user interface for the output of pathway prediction results.

The web page style sheet of the UM-PPS was initially written as a CSS file by computer science graduate student Ted Sands when he was working on the UM-BBD. This CSS file has been updated and is used by all the UM-BBD and UM-PPS web pages to provide a unified style on the user interface.

Generation of Pathway Elements

A pipeline system is used to produce a multi-level prediction and to visualize the prediction results. First, the system produces a multi-level prediction based on a recursive method; next, the system formats the pathway prediction results and feeds them into a graph rendering engine; finally, the system displays returned pathway graphics within a HTML map [46] on a web browser. The UM-PPS JavaServer Pages and servlets control the interactions between pipeline processes within the system, as depicted in Figure 4-2.

Figure 4-2. UM-PPS pipeline system infrastructure.



To construct such a prediction, the UM-PPS automatically predicts consecutive metabolic intermediates until one of several endpoints is reached. Figure 4-3 illustrates the system flow, and Figure 4-4 explains this recursive method in more details.

Figure 4-3. Flowchart showing UM-PPS system.

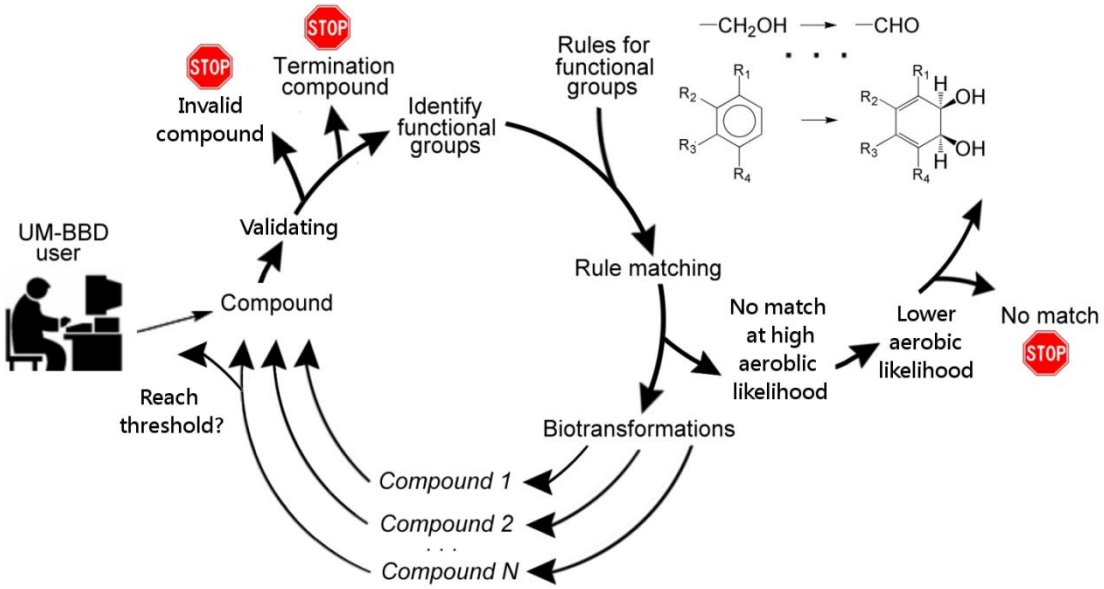
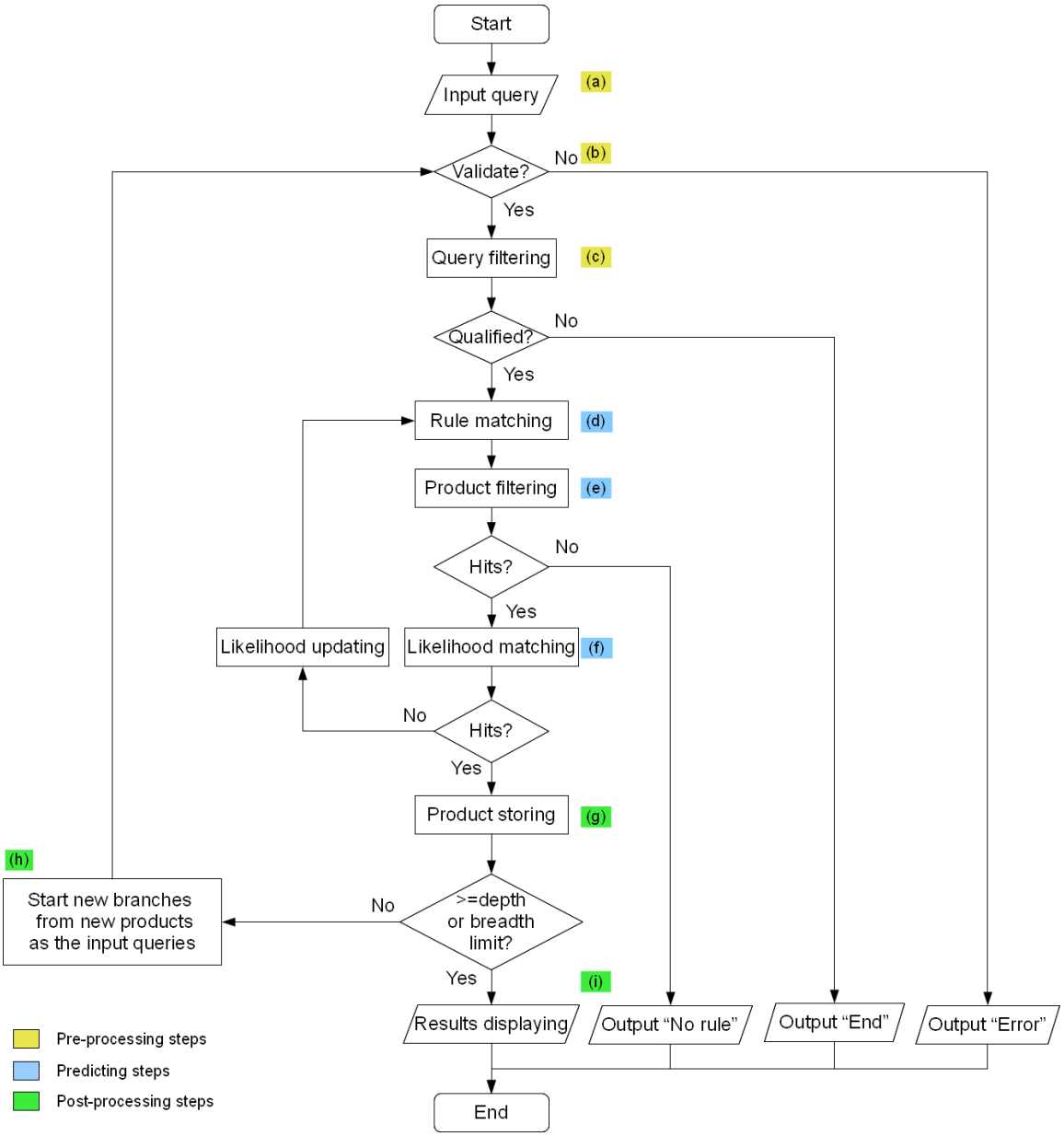


Figure 4-4. Flowchart showing UM-PPS multi-level prediction. (a) *Input query* takes a query compound and converts it into a SMILES string. (b) *Compound validation* checks the correctness of the string format and chemical structure, and terminates or hides predictions on initial submissions or intermediates respectively if incorrectness is found. (c) *Query filter* runs further checks on valid compounds, removing very low molecular weight compounds, pre-defined termination compounds, and some types of compounds that should not be predicted by the current version of UM-PPS [2]. (d) *Rule match* identifies functional groups in a query compound that match rule targets. If there is a successful match, a virtual transformation will be applied to the target functional group. (e) *Product filter* removes transformed products with fewer carbon atoms than a chosen cutoff value (default =3). (f) *Likelihood match* selects transformed products beyond a chosen aerobic likelihood value (either "aerobic" or "all"). If "aerobic" is chosen and there are no products, the UM-PPS will change to "all" and retry the prediction. If there are still no products, the prediction process will stop and a "No rule" message will be returned. At the end of a level, (g) *Product storage* merges products from all prediction branches and removes duplicates. If the total number of products at a level does not reach the chosen breadth cutoff value and the current level does not reach the chosen depth cutoff value, (h) *Level iteration* starts a new prediction branch for each transformed product and moves the prediction process into the next level. If either of these two cutoff values is reached, the UM-PPS will complete the prediction, and (i) *Results display* displays all products and pathways in a DAG.



The above method produces a multi-level prediction by submitting query compound against rules, iterating over all predicted products, testing each of their qualifications according to the filter settings, and displaying all qualified products in one or more plausible pathway branches. This procedure includes several steps in three categories: pre-processing (Fig. 4-4a-c), predicting (Fig. 4-4d-f), and post-processing (Fig. 4-4g-i).

Structuring of Pathway Elements

Pathway-like structures are not easily represented by using basic web languages, such as HTML and CSS. Those structures can be rendered quickly as a single diagram using GraphViz software [13]. The UM-PPS translates all elements of prediction results to the GraphViz dot language and feeds them into the GraphViz DOT program. Predicted products and biotransformation rules are converted to nodes and edges, respectively.

The ChemAxon Reactor will output all transformed products as SMILES strings for a triggered rule after each prediction cycle. The UM-PPS will assign a unique internal identifier for each product and store all products in a list with triggered rules and aerobic likelihood values (see Chapter I). A unique list will be created and stored in the global session zone of the Tomcat server at the end of each prediction step.

Pathway nodes and edges are extracted from the product list and formatted in GraphViz DOT language. Nodes and edges are coded in HTML-like labels. The predicted information of a node is coded in a TABLE element. A special point-shaped node constitutes the joint of a two-tailed arrow. A special undirected edge constitutes the head of a two-tailed arrow.

Representation of Pathway Structure

A node contains information on its chemical structure, including a SMILES string [5] and a 2D molecule graphic generated by ChemAxon Marvin tools [24]. A node may contain data source information if a product can be found in the UM-BBD or a subset of the KEGG PATHWAY Database [47]. A node may have multiple inputs if a product constitutes an intermediate found in more than once in a predicted pathway, and may have multiple outputs if it triggers more than one biotransformation rule.

An edge represents a biotransformation rule, containing information on its data source and biotransformation likelihood. Five different edge colors (Green, Yellow-green, Yellow, Orange and Red) represent five likelihood values respectively. An edge only has one head pointing to the substrate of the biotransformation but may have two

tails pointing to two co-products if that edge represents a biotransformation rule for a reaction that cleaves a single compound into two compounds.

The pathway structure is constructed through a recursive prediction method that starts from the initial query compound to further prediction levels according to whether the breadth or depth reaches the chosen cutoff values. When the prediction is complete, the GraphViz dot program will apply an automatic layout algorithm and render the graphical pathway.

Pruning of Graphical Pathway

A multi-level prediction may contain many duplicates of products and reactions. We added code to only store the first occurrence of a product, so that the UM-PPS can prune duplicates that start redundant branches.

Some of the biotransformation rules are cleavage rules that may produce two products at a time. Instead of listing all co-products individually, we group these products together to prune redundant edges. We designed a special arrow, labeled with a rule ID at the arrow head, and connected to co-products at the arrow tails.

The size of predicted pathway may keep growing when the prediction process continues. The increase of the predicted pathway may include some branches starting from compounds that users will not select as interested intermediates. We prune branches ending with these unselected nodes in the next prediction step.

The UM-PPS often predicts considerately smaller molecules that appear as common products of reactions. Too many arrows pointing to these products may increase the complexity of the graphical pathway. We also prune these compounds by default unless their molecule size is similar to their parents. Users may increase or decrease the size of compounds pruned.

RESULTS

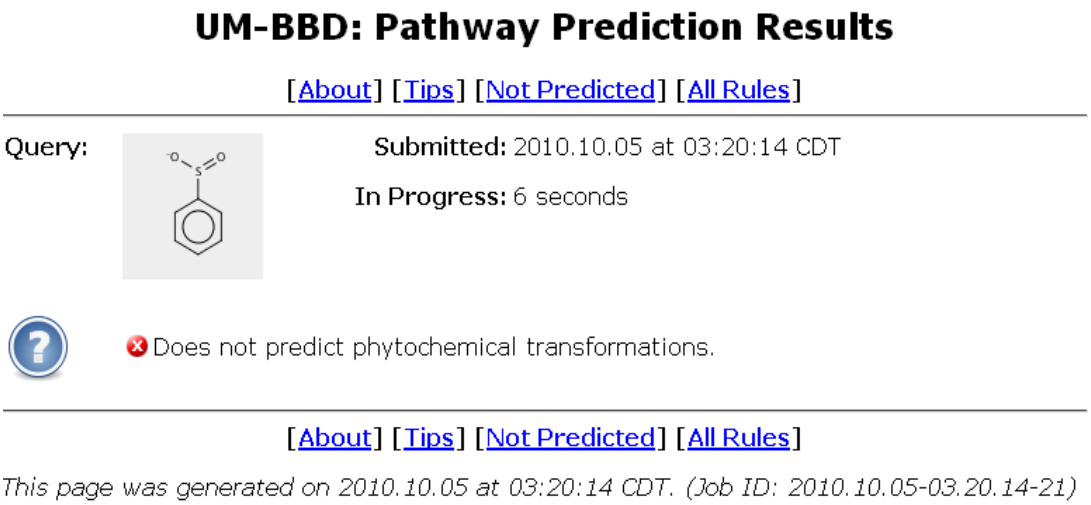
Graphical Visualization Tool

The new UM-PPS version has been under development since August 2009, and it was made available to the public in August 2010. The new system keeps the same home page and query options as the original system.

Because predictions, by default, run in aerobic mode, only Very Likely, Likely and Neutral transformation reactions are invoked and thus appear on the predicted graphical pathway. Predictions will show up to six levels in the first screen, display products containing more than three carbon atoms and stop at any level where there are more than ten products. These default options are changeable throughout the prediction process.

The UM-PPS shows more prediction levels than the original system, but its speed may be slower. We developed a status page that will be displayed to the users after they input a query compound and click on the "Continue" button. The status page is depicted in Figure 4-5.

Figure 4-5. The UM-PPS Status Page.



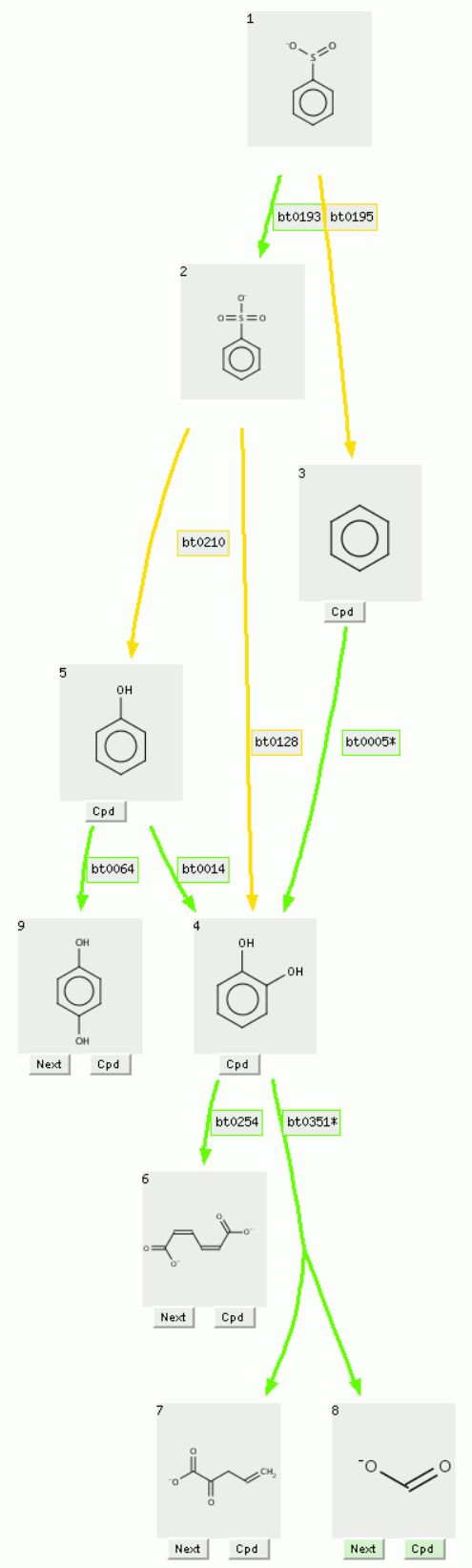
On the status page, the query compound is shown in a grey square in upper left of the status page. The query submission time and the prediction run-time are shown next to the compound square. The prediction run-time is refreshed once a second. A system usage tip is displayed under the compound square. Different random tips are shown every five seconds. A list of all tips is available (http://umbbd.msi.umn.edu/servlets/ppstips.jsp). A unique job ID is shown at the bottom of the status page. When the initial prediction on the query compound is finished, the system will bring a user to the Pathway Prediction Results page.

The UM-PPS has been successfully tested in Internet Explorer 8 [48], Firefox 3 [49], Google Chrome 5.0 [50], Opera 9.62 [51] and Safari 3 [52] browsers working on Windows XP/Vista/7, Ubuntu Linux 8.04 version [53] and Mac OSX 10.5 [54] of Intel architecture.

Aesthetic Metrics

The UM-PPS visualization uses the automatic graph drawing method of the GraphViz rendering engine [13]. By default, its automatic graph layout algorithm will minimize node displacement, node overlaps, edge bends, and the total edge length. The default global optimization does not contain local graph features like node level information that is very important for users to understand the prediction results. For example, another prediction results page for benzenesulfinate was shown in Figure 4-6. The default settings simplified the graph by only showing downward direction transformations but the prediction level information was not available. We changed the default layout to add node level information (Figure 4-7). We compromised some aesthetic standards for a larger total graph area but better usability.

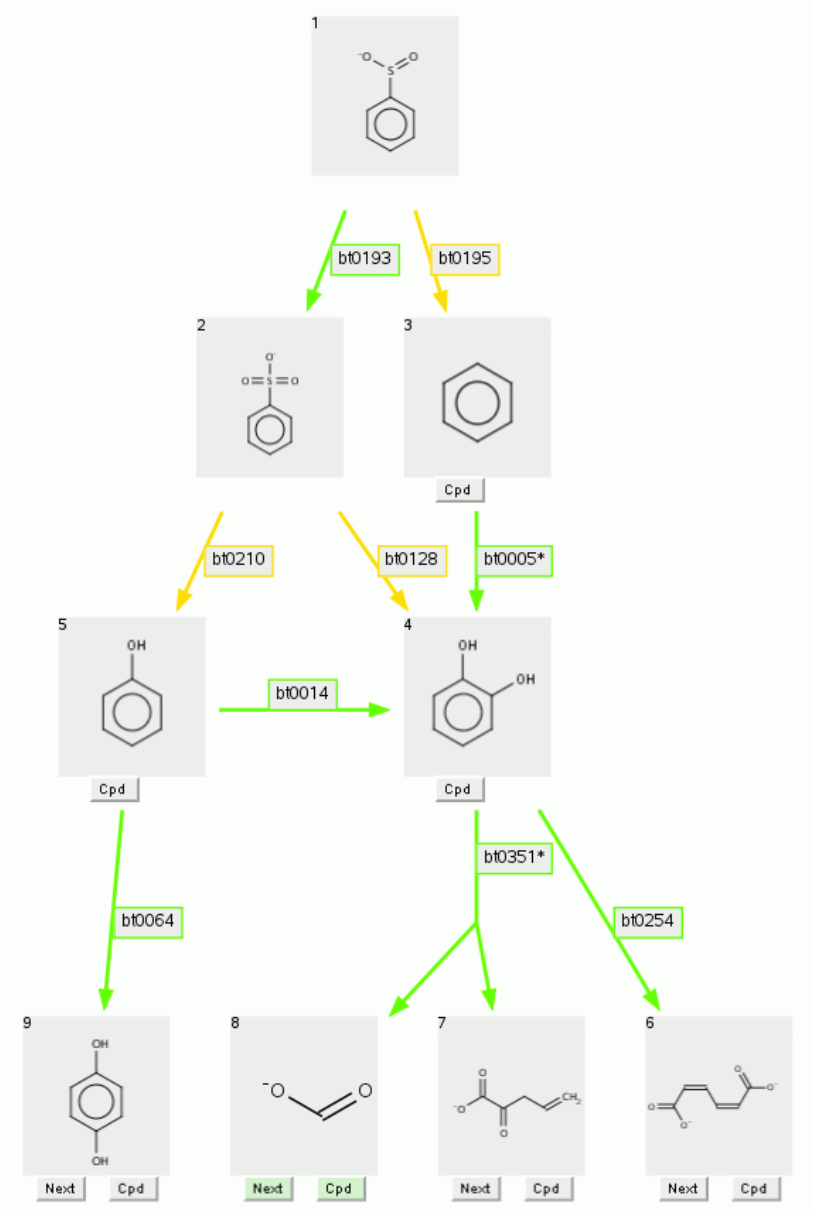
Figure 4-6. Prediction results for benzenesulfinate without level information.



Usage

To demonstrate and explain the implemented features, we analyze a predicted pathway for benzenesulfinate (C1=CC=C(C=C1)S(=O)[O-]), a compound not found in the UM-BBD, changing the default settings to show three levels and all products containing at least one carbon atom.

Figure 4-7. Three-level prediction results for benzenesulfinate.



As can be seen in Figure 4-7, there were a total of nine products, nine biotransformations, and seven pathway branches presented in a DAG. A product is shown

in a grey box, a biotransformation rule is shown as a colored arrow, and a pathway branch is constructed from contiguous compound boxes and rule arrows. Each compound box is labeled with an order number on its upper-left corner.

A common intermediate, catechol (compound 4 in Fig. 4-7) had three incoming biotransformations from three parent compounds: benzenesulfonic acid (compound 2 in Fig. 4-7) and benzene (compound 3 in Fig. 4-7) were shown at an upper level, and phenol (compound 5 in Fig. 4-7) was shown at the same level. Catechol also had two outgoing biotransformations pointing to three child products at a lower level. One of them was a cleavage biotransformation indicated by a two-tailed arrow, showing catechol being transformed to 2-oxopent-4-enoate (compound 7 in Fig. 4-7) and formate (compound 8 in Fig. 4-7).

The predicted pathway graphic is embedded in an HTML map. From the pathway prediction results page, one can go to the UM-BBD compound page when there is a "Cpd" button presented under a compound box (compounds 3-7 and 9 in Fig. 4-7). One can go to the KEGG database for metabolism of common compounds when there is a green "Cpd" button (compound 8 in Fig. 4-7). One can go to the UM-PPS biotransformation rule page by clicking on the edge labels. One can click on "Next" buttons (compounds 6-9 in Fig. 4-7) to choose a pathway branch to continue the prediction. All other branches will be pruned in the next screen.

To protect the privacy of information submitted to the UM-PPS, we added code to delete all predicted results as soon as users close their web browsers. We moved all submitted parameters from the web browser's URL to a wrapper page to provide a user-friendly interface. We developed an online tutorial page to guide new users (http://umbbd.msi.umn.edu/predict/aboutPPS.html).

System Evaluation

Before the new visualization tools was made available to the public in August 2010, we conducted a beta test using 12 volunteer UM-BBD users from different countries and research areas. The objective of the beta test was to explore the new system's functionality, accessibility, and stability. We included the survey description and results in Appendix A.

Seven beta testers completed the user survey. On the client operating system, six testers used Windows and one tester used Macintosh. On the client web browser, four testers used Internet Explorer, two used Firefox, and one used Google Chrome. All testers had used the UM-BBD and UM-PPS at least monthly in the past.

All items tested were rated by users on a 5-point Likert scale measurement [3] ranging from "very much disagree" (1 point) to "very much agree" (5 points). They agreed or very much agreed that the system is easy to use (4.5), the compounds on the screen are easy to see (4.6), the prediction layouts are easy to understand (4.5), the removal of duplicate products is important (4.0), and the indication of cleavage products is important (4.4).

The lowest rated survey item was the print function (3.6). Some testers complained that they had difficulties printing an over-sized pathway graphic. To improve this function, we added a Zoom feature that permits easier viewing and printing of prediction results. One beta tester tested the Zoom feature and agreed that he was now satisfied in how predictions looked when they were printed (4.0).

DISCUSSION

The rule-based UM-PPS now provides a tree layout for biodegradation predictions. An interactive Pathway Prediction Results page allows users to examine any predicted products at any position in a metabolic tree, and to save web page for local viewing. This new interface holds great promise for a better visualization of biodegradation prediction results. Because biodegradation pathways reflect an extraordinary degree of complexity, however, visualization alone does not present an entirely comprehensive model of prediction results. Yet by basing additional studies on the DAG structure, which offers considerably more potential for further insight into prediction results, we may arrive at a more complete model with which to stimulate future analyses.

Interoperability

The current UM-PPS only accepts query compounds submitted as SMILES strings. It may increase system interoperability if the UM-PPS can convert and accept

other widely used molecular file formats, such as SDF [55], MOL [56], and MDL [57], etc. The UM-PPS only outputs the predicted results using an HTML map on a web page. A function of exporting the results to other file formats may allow users to visualize the pathway using other software platforms, such as Cytoscape [58], to conduct further analysis. The UM-PPS runs as a web-based tool and provides a querying web interface. Besides this interface, providing some APIs (Application Programming Interfaces), such as SOAP [59], may allow the UM-PPS to be integrated by other bioinformatics systems.

Zoom ability

The current Zoom feature cannot support unlimited zooming in or zooming out without pixelization. One UM-PPS developer suggested using the SVG format [60] presenting the graphical pathway to provide a smooth zooming effect. Since, by default, the SVG format is not supported by the IE web browser that is used by more than 40 percent of UM-PPS users, we decided not to use it in the current version but may consider implementing it in the future.

Interactive feature

The current system doesn't allow users to delete or hide any node from the results set, one feature requested by a beta tester. It may improve the system interactivity by implementing this, so that users can simplify or customize the pathway view.

CONCLUSION

The need for expert prediction of biodegradation pathways has driven the improvement of the UM-PPS visualization to simultaneous depict multiple intermediates and cleavage products. Users can now view prediction alternatives much more easily. The system fully supports existing metabolic logic entities, and it can produce a multi-level prediction and display the results in a DAG. Beta testing users were satisfied on its functionality of visualization.

Chapter V

IMPROVED UM-PPS PERFORMANCE

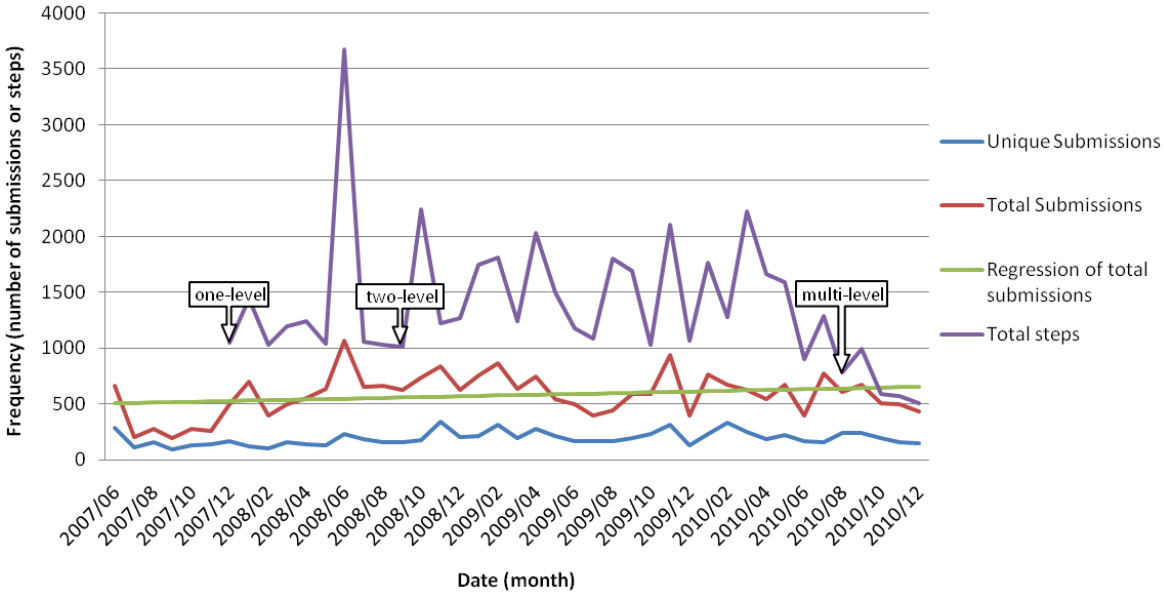
INTRODUCTION

The UM-PPS is a widely recognized pathway prediction system used to improve bioremediation [61]. Predictions are based on biotransformation rules that, in turn, are derived from reactions found in the UM-BBD and the scientific literature. Since the system was created in 2002, its biotransformation rule base has grown to almost 280 entries. The continuous growth of the UM-PPS knowledge base increases the importance of performance improvements.

Current system usage

Each UM-PPS submission frequently takes several consecutive steps to complete. A step is the process to complete one or more levels of prediction at a time, and it shows one, two, and up to six levels of prediction in one-level, two-level, and multi-level system, respectively. We started to capture submissions in May 2007 and prediction steps in December 2007. From June 2007 to December 2010, the UM-PPS processed over 20 thousand su bmissions, and over 600, up to 1069, submissions per month. As depicted in Figure 5-1, the blue line shows the number of unique submissions per month, the red line shows the number of total submissions per month, and the purple line shows the number of total prediction steps per month. The green line shows the linear regression of total submissions that indicates an overall 9% annual increase in system usage.

Figure 5-1. The usage of one-level, two-level, and multi-level UM-PPS



The one-level UM-PPS had been the only available tool until two-level UM-PPS was released in August 2008. Both tools were available then, and the one-level system was more widely used as the default choice. In August 2010, the multi-level UM-PPS was released, and it replaced the original versions. The multi-level UM-PPS increased its prediction ability by predicting more levels per step, and users tend to complete a prediction in fewer steps than before.

The demands of performance improvement

The original UM-PPS ran predictions using a serial strategy. The system ran the user's submitted compound against one biotransformation rule at a time, and ran as many times as the total number of rules in a prediction cycle. With the increase of UM-PPS rule-base, a prediction cycle may take more time to complete. It takes approximately 10 seconds to complete the initial step of the prediction for demo compound, Benzyl Alcohol (OCc1ccccc1), in the original UM-PPS. If the system predicted a complete biodegradation pathway in more than one step, the run-time would grow exponentially. Based on archived statistics, about one third of predictions took more than one step to complete. The longest prediction was completed after 43 steps in Jan 18, 2008. We started to develop a multi-level UM-PPS in August 2009. The time-cost on a multi-step prediction may not be affordable for an online application. Performance improvements for such a computing-intensive system are imperative.

The UM-PPS is a web-based biodegradation prediction tool that performs prediction tasks requested by clients using a web browser. However, the original system does not have the capability to process simultaneous accesses at the same time: when the system is busy on one prediction task, another later task will not start until the previous task is completed. If incoming tasks pile up on a busy system, the first user will block other users and cause queuing interference. In 2009, the UM-PPS processed 7,394 submissions consisting of 18,279 prediction steps. Competitive prediction steps tended to cause queuing interference. Thus, it is important to evaluate system usage to determine the extent of this, and determine how to reduce it.

System usage can be measured by two parameters: the run-time variance of public system submissions and related submission timestamps. In a simulation study, we ran the

public system submissions against the multi-level UM-PPS and produced the run-time distribution. Based on the run-time distribution, a statistical model was built and used to generate random run-time at each prediction step starting from a given timestamp. All prediction steps were analyzed and their overlaps were used to simulate queuing interference. The percentage of queued prediction steps was estimated.

Web crawlers can degrade the UM-PPS performance by automatically sending submissions. Based on the server logs, a great amount of visits from web crawlers were observed from June to September in 2010. In August alone 2,000 out of 2,610 total visits were observed from Google search engine. Such visits will not only waste computing resources, but also increase the risk of queuing interference. Thus, it is important to prevent web crawlers from triggering the UM-PPS.

METHODS

Hardware and software

The current UM-PPS runs on an Intel XEON server with a Linux 32 bit operating system hosted in the Minnesota Supercomputing Institute (MSI). The server's 4-core processors permit potential performance improvements by using concurrent programming approaches. Additional computing resources are available to increase the speed of an individual prediction by using multiple threads, and to reduce queuing interference caused by competitive visits to the prediction tool.

The procedure for optimizing prediction performance requires determining the number of threads providing the best performance, and reducing performance variance of the MySQL and Tomcat servers. The procedure for reducing queuing interference requires determining the frequency of public system submissions, simulating queuing interference, and reducing system abuse.

Perl [40] was used to extract public system submissions and timestamps from the Tomcat server log file; Java [35] was used to run batch predictions of user entered compounds in the multi-level UM-PPS; R [62] was used to develop a statistical model of run-time distribution; and MS Excel was used to simulate queuing interference.

Determining the number of threads

The UM-PPS core is coded in Java and runs in a Java Virtual Machine (JVM) which is capable of parallelizing the work-load on multi-processor architectures. Java programming framework provides an Application Programming Interface (API) for scaling Java web application performance by breaking the computing resources down into multiple parallel threads.

To better utilize the multi-core CPU, a parallel strategy was used to divide a prediction cycle into various sub-tasks in two ways. The strategy can be implemented in either serial fashion or cyclic fashion. Each sub-task includes an allocated number of predictions on corresponding biotransformation rules. All sub-tasks run concurrently, and predicted results are merged together when the last sub-task is done. However, with the variation on the number of sub-tasks, under-utilization or over-utilization of CPU resources may occur. Therefore, a performance test was conducted on determining the number of sub-tasks which result in optimal utilization.

The computing performance was measured by the prediction run-time of Benzyl Alcohol (OCc1ccccc1) against all UM-BBD biotransformation rules for its first prediction cycle. During a prediction cycle, the substrate, Benzyl Alcohol, would be paired with each rule and match four of them, producing four products. The same data set ran on a sequence of computing threads $\{2^n\}$, where n = 0 to 8, and a special thread range from 2 to 8. The threads divided the whole workload evenly into multiple sub-tasks of their own. The JVM acts as a black box to schedule threads on physical CPU cores and to synchronize threads without manual intervention.

Reducing performance variance of MySQL and Tomcat servers

The UM-PPS web server has installations of MySQL (version 5.0.26) and Apache Tomcat Server (version 5.5.28). Prior to modeling real system usage, the supporting software packages were tested and configured to provide optimal and stable performance.

First, we developed code to reduce the MySQL server variance. The UM-PPS has experienced connection refused problem in the past when new connection requests were sent to a fully loaded MySQL database. This problem was found to be related to MySQL configuration parameters: **max_connections** and **wait_timeout**.

The variable **max_connections** stands for the number of simultaneous client connections allowed. The MySQL database can receive intensive queries within a short time window. For example, around 13,000 queries were received with a 5 to 30 seconds interval on October 17, 2009. Official MySQL documentation recommends setting **max_connections** to at least 100 to optimize performance [63].

The other variable **wait_timeout** stands for the number of seconds the server waits for activity on a non-interactive connection before closing it. The default value of variable **wait_timeout** was set to 28,800 seconds (8 hours). Since the MySQL server serves the UM-PPS as a web application database instead of a data center, it is not necessary to allow idle connections for 8 hours. The default setting increased the risk of using up the available connections in a short time. According to a user report in a MySQL forum [64], lowering **wait_timeout** will help prevent hitting the threshold of **max_connections**. Another report stated that a MySQL connection was unexpectedly lost when **wait_timeout** was too low, and suggested an optimal value of 120 seconds [65].

Second, we developed code to reduce database queries by caching the UM-PPS knowledge base in the server's memory. In the original UM-PPS, a prediction cycle would query several tables in the MySQL database more than one time. If the UM-PPS can access the data from the server's memory without creating new database connections, it will speed up the prediction process. The Tomcat global session objects stay in server's memory and permit hosted applications to access their variables during the uptime. We use global session objects to store the UM-PPS knowledge base that includes all UM-BBD compounds, all UM-PPS biotransformation rules and five kinds of metabolic logic entities.

Third, we developed code to reduce Tomcat server variance. The UM-PPS had a large run-time variance when predicting the same compound at different time. Sometimes the "Out of Memory" error message even was given to a 'short' prediction when the UM-PPS was being debugged. We consulted MSI and server status was retrieved from the Tomcat log file. No exceptions, but some reloading events, were found among archived prediction records. Since the Tomcat server is hosted in a JVM, the JVM statistics can reflect the Tomcat performance changes. To understand the time variance problem,

related JVM variables were monitored. A small compound, Pyrrole (c1cc[nH]c1), was chosen as a probe in run-time testing during the monitoring period.

All test runs collected server variables and probe run-times every 30 minutes for 200 time points. The first test started on December 10th and ended on December 14th, 2009. The second test was conducted on a freshly restarted Tomcat server without development activities and reloading interference. This test ran from December 15th to December 19th, 2009.

Determining the time variance of public system submissions

First, we developed methods to determine the sample size for time-variance of public system submissions. We used the 12 months data of 2009 to study the frequency of public system submissions. There were over 7 thousands raw records and around 15 hundred unique records in the selected data set. Perl scripts were developed to remove incorrect records of illegal SMILES strings, valence errors or absence of C atoms, and records of compounds that are too small to be predicted in the system.

The probe compound, Pyrrole, was studied to determine a sample size for timing reliable run-time ranges of real system submissions. 200 time points from Dec 15, 2009 to Dec 19, 2009 were used. We assume that these points represent the real distribution of the probe compound's run-time, and we used a statistical model to determine a sample size or a number of timing points could give a good estimation of the real distribution.

A two-tail t-test was used to evaluate the H_0 hypothesis that the probe compound's run-time is not equal to selected samples' average run-time. Five different sample sizes of 5, 10, 15, 20 and 30 were chosen. For each sample size, samples were randomly selected from 200 points for five times. Samples' average and standard deviation values were calculated each time. These values were tested against the probe compound's average run-time, 2.02 seconds within its 95% confidence interval. The following equation was used:

Equation 5-1: $$\text{Power} = P\left(\frac{|\bar{x} - 2.02|}{\sigma/\sqrt{n}} \le 1.96 \mid H_1\right)$$

Where $\bar{x}$ is sample average run-time, σ is standard deviation of the sample run time, n is sample size, 1.96 is the approximate value of the 97.5 percentile point, and H_1 stands for the hypothesis that the probe compound's run-time is equal to selected samples' average run-time.

Second, we develop code to determine time-variance of public system submissions. We acquired a new server from the MSI with the same configuration of current UM-PPS. This independent server allowed us to time public users' submissions without interrupting the production servers. Batch processing scripts were developed to run each submission one at a time, for 30 repetitions. Predictions ran in aerobic mode so only very likely, likely and neutral transformations were shown. Predictions would show up to 7 levels in the first step, display products containing more than 3 carbon atoms and stop at any level where there were more than 10 products.

Simulating queuing interference

When a user submits a compound to the UM-PPS that is busy, the new prediction request will be queued until the previous prediction is finished. Since queuing interference will increase prediction run-time, it is important to understand its occurrence frequency.

First, we developed code to extract 18,279 submission timestamps from 2009 data set. We used all submission records that were archived at the initial prediction step. Each record contains the information in submitted compound, prediction step, timestamp, and client IP address. We only used the timestamps field in this simulation study.

Second, we developed code to generate random run-time data set from the run-time distributions of 2009 data set. Based on the study of time variance of public system submissions, we need to understand the statistical distribution of these submissions to simulate the queuing interference. We calculated the average run-time, and generated a Q-Q plot to compare the quantile of the run-time data with the quantile of a generated log-normal distribution based on the average run-time [66]. If we can observe a linear relationship between these two quantiles, we can use the log-normal model to generate random run-time data at each timestamp, and simulate the queuing interference. The following probability density function of a log-normal distribution was used:

Equation 5-2: $$f_X(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}}\, e^{-\frac{(\ln x-\mu)^2}{2\sigma^2}}, \quad x > 0$$

Where μ and σ are the mean and standard deviation of the log of run-time data.

The simulation results will be used to identify the number of additional servers to solve the queuing interference issue. After the implementation, we used the queuing data extracted from the public UM-PPS server logs to further verify the simulation results.

Reducing system abuse

The Robot Exclusion Standard, or *robots.txt*, is a protocol to prevent cooperating web crawlers from visiting all or part of a website which is publicly viewable [67]. The *robots.txt* file contains **User-agent** and **Disallow** fields to specify the scope of exclusion. The **User-agent** field defines specific web crawlers, and the **Disallow** field defines file locations that web crawlers are not allowed to visit. Since the robots.txt is an advisory protocol, the UM-PPS developers only need to add certain UM-PPS URLs into the *robots.txt* file instead of adding additional functions to the current system.

RESULTS

Four threads provides the best performance

Figure 5-2. Multi-thread performance histogram

The first test produced (a) a multi-thread performance histogram on a sequence of threads $\{2^n\}$, where n = 0 to 8; the second test produced (b) a multi-thread performance histogram on a thread range from 2 to 8.

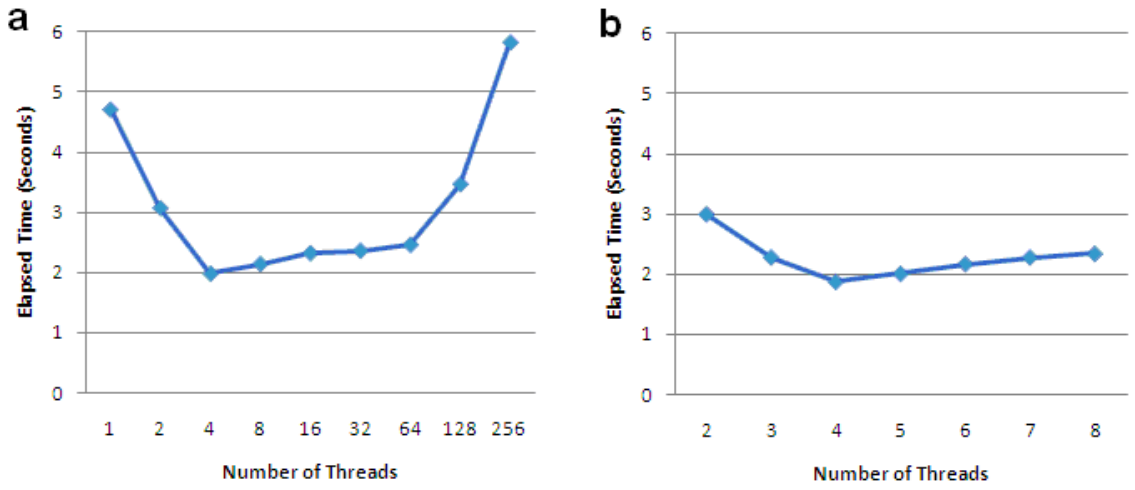


Figure 5-2a is a histogram showing the distribution of run-time on different numbers of computing threads. The run-time stands for the period from the beginning of the prediction to the point when the predicted result is given. The run-time is the average value taken from 10 predictions on each thread number.

Run-time first decreases and later increases as the number of threads increase. It reached the lowest point, 1.98 seconds, when the test ran on four threads. As depicted in Figure 5-2b, the lowest run-time was also seen a test that ran over a thread range from 2 to 8 when four threads were used. Thus, four threads were determined to provide the best performance on a 4-core CPU server.

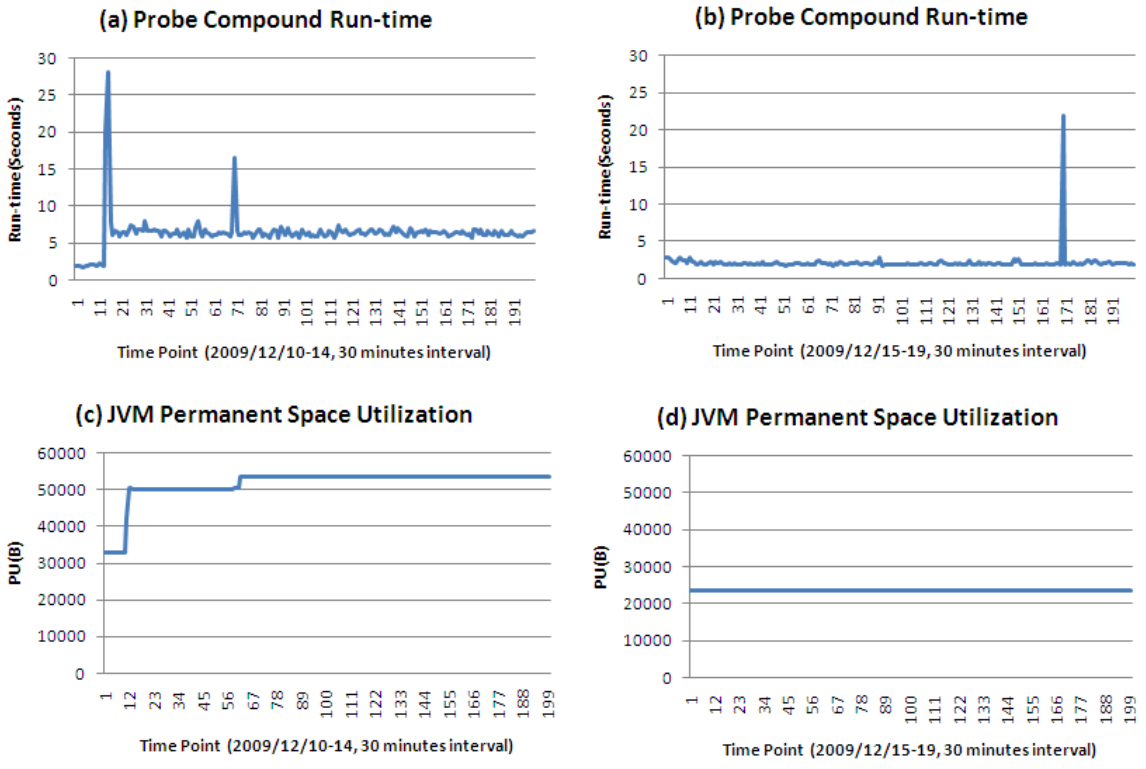
We developed a multi-thread programming strategy by using four threads. During a prediction cycle, all rules are divided evenly to four groups, and each group runs against the submitted compound in its own task. These four sub tasks are starting concurrently, and all predicted results are merged together when the last sub task is done. Since different numbers of rules may be triggered in each sub task, the workload of a sub

task is not always equal to each other. This explained the fact that performance is only twice, not four times, as fast.

Changes on MySQL and Tomcat servers

MySQL server performance was stabilized. The UM-PPS experienced unexpected system downtime twice in 2008 and four times in 2009 due to MySQL "too many connections" error. Upon our request to the MSI, the **max_connections** was increased from 30 to 100 concurrent connections, and the **waiting_timeout** was changed from 28,800 to 120 seconds. These changes allow the database to handle heavy loads at peak times and clean up idle connections that have been in non-interactive status for more than two minutes. After these changes, the "too many connections" problem did not occur during the next 12 months.

Figure 5-3. JVM monitoring and probe compound run-time charts. The first test produced (a) a probe compound run-time chart and (c) a JVM permanent space utilization chart. The second test produced (b) a probe compound run-time chart and (d) a JVM permanent space utilization chart.



Tomcat server performance was stabilized. Figure 5-3a shows that, during the first test, the probe run-time started from 2 seconds and increased to 6 seconds after a big spike at time point 15. A smaller spike was seen at time point 70, but run-time remained stable then. A similar trend was found in a server variable, PU (Permanent Space Utilization), in the same time frame (Figure 5-3c). Two jumps, which happened at the same time points seen in Figure 2a, changed the PU from 32 megabytes (MB) to 53 MB, as shown in Figure 5-3c.

The time points 15 and 70 were debug periods when we were recompiling Java classes, and the Tomcat server was reloading the updated web application. Since the Tomcat server refills new classes definitions into its permanent generation space during the reloading process, if memory leaks occur, the reloading behavior could be one of the major reasons for the variable PU and probe run-time in the first test.

In the second test, both PU and probe run-time stayed stably around 25 MB and 2 seconds respectively depicted in Figure 5-3b and 6-3d. Figure 5-3b showed a run-time peak at the time point 170. Since the probe was following other predictions in a queue, this incidence didn't account for the run-time variance.

The above observation suggested that when the JVM permanent space utilization approached its original maximal value of 64MB, the probe compound run-time increased. Thus, the JVM permanent space was confirmed to be a factor that can affect the Tomcat server performance. To standardize the prediction run-time, the allocated permanent generation space was requested to be increased from 64 MB to 256 MB based on the Java Tuning White Paper [68]. The UM-PPS software deployment strategy was updated to require a Tomcat restart after any major code change.

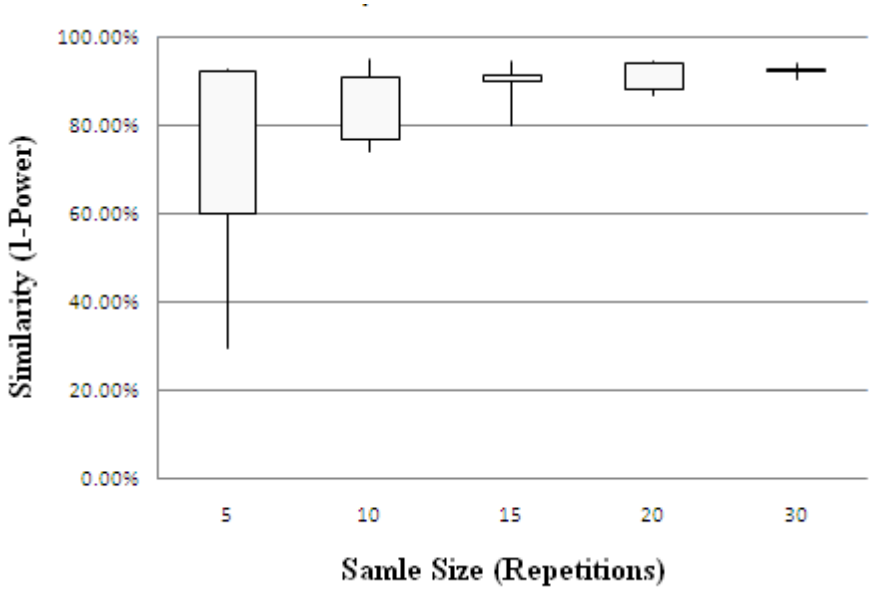
Although the reasons for the server overload are clear, it is still useful to monitor the server parameters to reduce performance variance. We developed monitoring strategy that periodically monitors the number of processes and the size of Perm Space on Tomcat servers, and the number of connections on MySQL database server. We developed code to check server status once an hour in daytime (from 6 AM to 6 PM) and once every two hours in evening (from 6 PM to 12 AM). Exceptions are reported when the monitored parameters exceed the defined cutoff values. We also developed a system test strategy for all functionalities of the UM-PPS, so that depreciated APIs will be detected without

causing warnings and exceptions after each ChemAxon software upgrade. When such events happen, the UM-PPS developers will receive a report so that they can restart the servers promptly. These improvements increase the precision of timing public system submissions.

Time variance of public system submissions

We first determined the number of timing repetitions to produce precise run-time data. Statistical power values were calculated over five different sample sizes of repetitions. Large power value means there is a good chance to support the hypothesis that probe compound's run-time is not equal to selected samples' average run-time. So we use the value, *1-Power*, to measure the similarity of distributions between the probe compound's run-time and selected samples' average run-time. The test results were depicted in Figure 5-4.

Figure 5-4. Run-time similarity box-plot



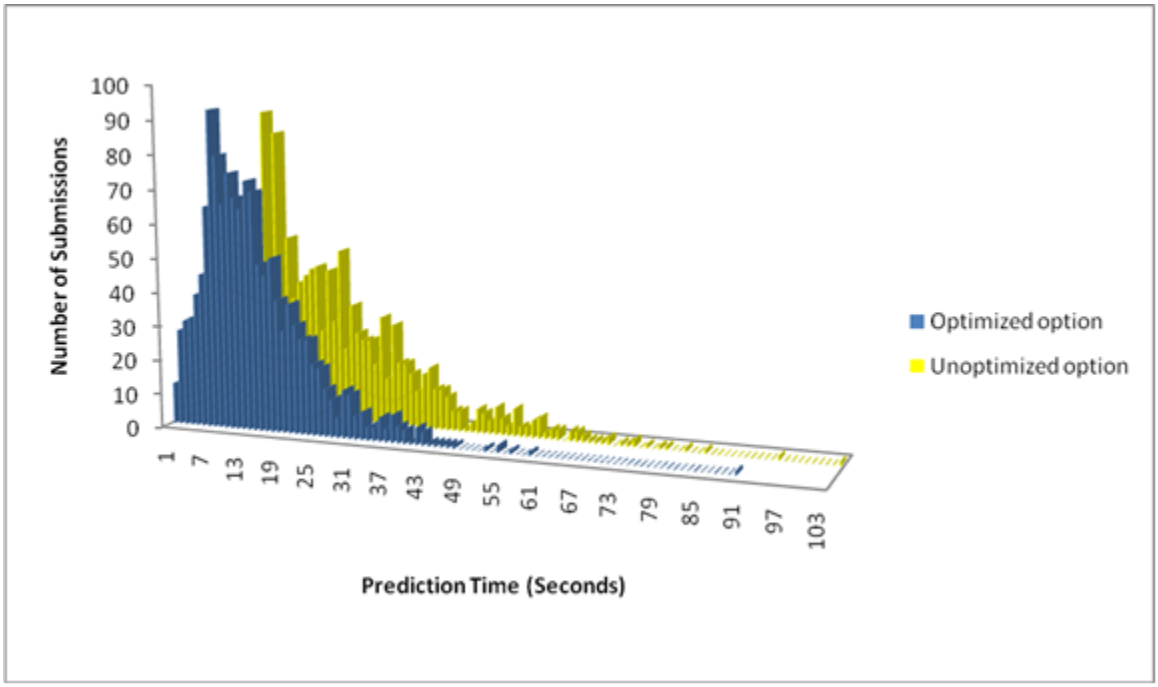
When sample size was small, the similarity was lower and spanned a wider range, indicating a good chance there would be a difference between the probe compound's run-time and selected samples' average run-time. When the sample size increased, the similarity increased. When 30 samples were selected, the samples' average run-time was more than 90% likely to be equal to the probe compound's true run-time. So 30 repetitions were used to time the public system submissions.

We removed duplicate compounds from the original data set and applied filtering scripts to 1,536 unique compounds. 25 incorrect compounds and 157 poor quality compounds were deleted. 12 of remaining compounds were redirected to "All" option since no biotransformation rule applied on them by default. They had a short run-time of less than 0.03 second, and these outliers were removed from the timing test. After the filtering, there were 1,332 compounds left in the test data set.

Based on the default prediction options described in the method section, each compound was submitted to the UM-PPS one at a time, for 30 repetitions. Run-time data was extracted from Tomcat log files. For each submission, its run-time was defined as the average run-time of 30 repetitions. The run-time data was analyzed using R Hmisc package [62].

Two different server settings were explored. One run-time test was done without cyclic thread balancing and knowledge base caching. Another set of test was done with the above two performance optimizations.

Figure 5-5. Real system submissions run-time distribution



In both settings the run-time distribution had a long tail on the right as seen in Figure 5-5. Thus the median is the better estimate of central tendency.

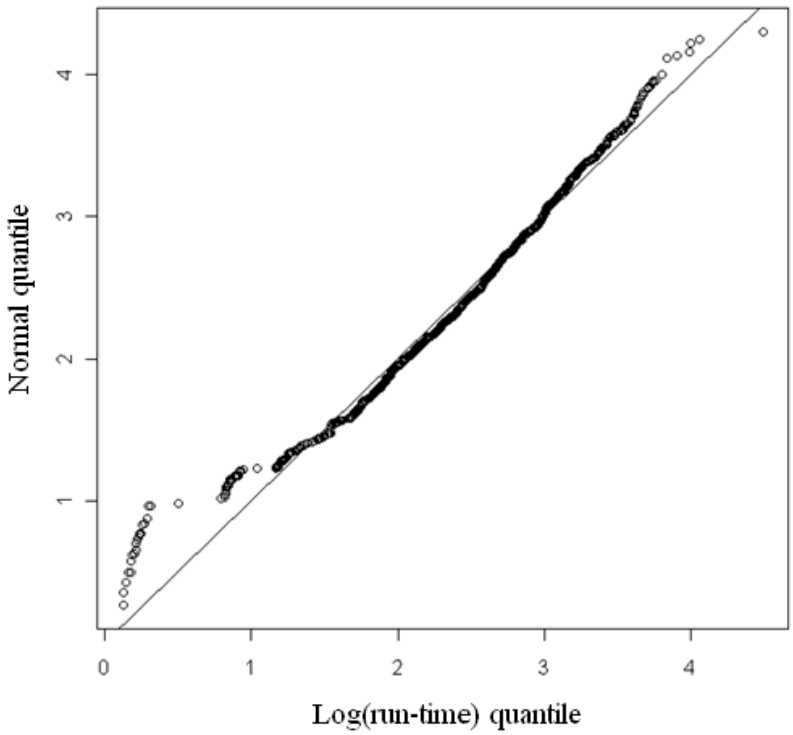
The blue bars show the run-time with cyclic thread balancing and knowledge base caching, while the yellow bars show run-time without these options. With these options, the run-time ranged from 7 to 18 seconds between 25% and 75% quartiles, the median run-time was 12 seconds, and the longest run-time was 89 seconds. Without these options, the run-time ranged from 11 to 30 seconds between 25% and 75% quartiles, the median run-time was 19 seconds, and the longest run-time was 103 seconds.

Based on the comparison, the optimization of cyclic thread balancing and knowledge base caching resulted in a more than 30 percent increase in prediction performance.

The simulation of queuing interference

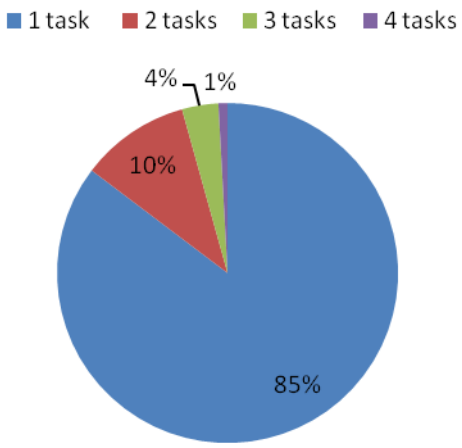
Based on the study of time variance of public system submissions, we studied the run-time distributions of 2009 data set, as depicted in Figure 5-5. We took the log of the run-time and plotted its quantiles against the normal distribution quantiles. The mean and variance of the normal distribution were estimated as the sample mean and sample variance of the log run-time.

Figure 5-6. Log-normal distribution Q-Q plot



From the Q-Q plot (Figure 5-6), we found that the log of run-time followed the normal distribution, thus the run-time followed a log-normal distribution. We extracted 18,279 original timestamps from 7,394 submissions in the 2009 data set. At each time point, a generated run-time from the log-normal distribution would be given. As depicted in Figure 5-7, all queuing events were classified according to the number of tasks in a wait queue.

Figure 5-7. Queuing interference classification



From the simulation results, 462 of 18,279 prediction steps (2.53%) were queued. The largest class, about 85% of the queuing interference, was formed by only one queued submission. In other words, if the UM-PPS had a second server that could process this number of queued submissions, the overall queuing interference frequency can be reduced from 2.53% to 0.38%.

This study suggests that two servers can significantly reduce the queuing interference at minimal cost. We acquired a second server for the UM-PPS and moved both servers to public in August 1, 2010. In the following three months, from August to December, the UM-PPS processed 3,437 prediction steps from 2,726 submissions. 70 of these steps were redirected to the second prediction system, 63 of them were processed immediately, and 7 were queued. The frequency of using the second server was 2.04%, and the chance of being queued was 0.20%. Overall, the observed redirection and queuing frequencies agreed with the simulation results, thus two UM-PPS servers solved the queuing interference problem.

Web crawler exclusion

We created a *robots.txt* file under the root directory of the Apache server. We used the wildcard "*" to specify all web crawlers in the **User-agent** field, and added two UM-PPS files, *predict.jsp* and *predict_wrapper.jsp*, in the **Disallow** field. Based on the exclusion settings, all cooperating web crawlers will not trigger the UM-PPS by visiting the above two files that act as the system interface. We manually removed all web crawler submissions from previous log files that were used in the simulation study.

DISCUSSION

Computing performance

The available computing resources determined the multi-thread performance. When the number of threads was small, threads tended to have sufficient computing resources. With the number of threads increased, threads tended to be in waiting state for available processors and might over-use limited computing resources. Based on our test, with the number of threads went away from four, either smaller or larger, prediction run-time increased.

The two UM-PPS servers lack a load balancer that, upon getting a submission, sends it to the server with the least number of submissions in queue. If the usage of the UM-PPS increases in the future, such a load balancer will be highly desirable.

Server monitoring

JVM statistics were monitored for 200 time points in a five-day time frame twice. The monitoring was sufficient for explaining the negative effects of class reloading events on Tomcat performance, but might not be sufficient to indicate that reloading factor is the only reason for the performance variances. To evaluate the complex server environments, more sophisticated tools, e.g., Java Profilers [69], may be required for dynamic performance analysis. Further analysis may provide a better solution to prevent the UM-PPS servers from experiencing unexpected downtimes caused by Tomcat server or MySQL database overload.

Simulation deficiency

12 months data of 2009 contains 1,332 valid records. Around 30% submissions are UM-BBD compounds and around 80% submissions are found in the PubChem database [70]. This data set is a representative sample of recent public UM-PPS usage. The timing statistics were based on the first-step predictions based on default options that are described in the methods section. If the UM-PPS changes default prediction options in the future, statistics resulted from this study will not be applicable to simulate real system usage.

The observed queuing interference frequency was smaller than the simulation results. This may be due to the over estimation of system usage. In the simulation study, we used the submission timestamps of 2009 date set that was generated by original one-step UM-PPS. The multi-level system predicts more results than the original system, thus users usually take fewer steps to complete a prediction now. We saw 2.47 steps per submission in average in 2009, but we only saw 1.26 steps per submission in the data set retrieved from August to December 2010. The decrease in the prediction steps per submission explained the difference between the simulated queuing interference frequency and the observed frequency.

Reducing system abuse

Using robots.txt is a way to tell web crawlers what not to crawl, but it cannot block malicious web crawlers that do not comply with this protocol, or they may mask their identity. An alternative way is to block web crawlers by using their IP address patterns. For example, all observed submissions from Google web crawlers used IP addresses starting from 66.249.67.1 to 66.249.71.206 [71]. Based on these observations, we can develop regular expression patterns to match such IP addresses and exclude them from using the UM-PPS. Another way is to identify the hostnames by converting the IP addresses. For example, IP addresses starting from 66.249.67.1 can be converted to hostnames based on the root domain name googlebot.com. The UM-PPS can recognize all sub domains, like crawl-66-249-67-1.googlebot.com, from web crawlers and reject their submissions.

Using Captcha [72] is another way to exclude robots by requiring users to type letters or digits from a distorted image that appears on the screen. However, the disadvantage of this approach is that the system will need additional manual intervention and may even be inaccessible to visually impaired users.

CONCLUSION

The UM-PPS has continued its growth in the past 8 years, and its influence increases with over 7,000 submissions per year. The original UM-PPS matched the submitted compound with all rules one-by-one in a prediction cycle. A multi-level prediction predicting many products at a time can be computationally intensive and thus requires the user to wait longer than desired for the prediction.

Now the UM-PPS is hosted at a high-end 4 CPU server, and the upgraded hardware permits improvement to the system's computing performance, using multi-thread programming strategy. We used a multi-thread computing method that decreased the overall prediction run-time by half [1]. We balanced the computing threads by assigning biotransformation rules to them in a cyclic fashion. Since the UM-PPS frequently queries data from the UM-BBD MySQL database, we pre-loaded all related tables to permit quick access to data. Both improvements resulted in an additional 30% decrease in overall system run-time. We acquired a second server for the UM-PPS that reduced the queuing interference frequency to less than 1%. These improvements lead to smarter and faster UM-PPS.

Chapter VI

CONCLUSIONS

Biodegradation pathway predictions are usually complex due to the diversity of biotransformation rules and multiple metabolites. A visualization approach that displays a complete metabolic network should help users both refine alternative choices and better understand prediction results. However, a multi-level prediction that produces such a metabolic network can be very time consuming. Thus, a computing strategy will be required to maximize the prediction performance and minimize the queuing interference with available system capability. Overall, improving UM-PPS visualization and performance is the primary goal of this study.

In the first phase, methods and software that could be used to improve the UM-PPS visualization and performance were explored. Selected biological data visualization software that can facilitate biodegradation network presentation was evaluated. Various visualization methods used by current biodegradation tools were discussed, and a set of features that can be used for a better prediction were identified. DAG format was selected as the graphical presentation for predicted biodegradation pathways, and GraphViz software [13] was selected as the graph rendering engine to produce DAG files. Current parallel computing methods were explored, and Java multi-thread infrastructure was selected. Performance optimization approaches like database caching and queuing simulation was described. The above approaches permit the implementation of better visualization and performance of the UM-PPS.

In the second phase, a two-level prediction and visualization approach was implemented. It increased the prediction depth from one level to two levels per prediction step. Thus, more predicted metabolites were displayed at a time. However, this approach lacks the capability of clearly identifying common metabolites and cleavage products, which are important for an easy understanding of the prediction results. Additionally, this approach could only show complete metabolism of a substrate that has no more than two levels of products. Since predictions usually take more than two levels to complete, users are still required to have extensive expert knowledge to make educated choices. Two-level visualization approach uses different layout styles to display the first level of products and the second level of products, and it is not extendable to display further levels of products; therefore, new visualization approaches need to be explored.

In the third phase, a multi-level prediction and visualization approach was implemented. This approach uses a new recursive method to produce a biodegradation network and uses GraphViz graph rendering engine to output graphical results. It is able to show complete metabolism of a substrate that has up to seven levels of products by using an interactive HTML map entity. The visualization interface can display over one hundred metabolites and transformations at a time, and it can clearly identify common intermediates and cleavage products by using multi-head arrows and multi-tail arrows respectively. This approach uses DOT[13], a standard plain text graph description language, to render graphical files, thus it can be easily extended to display more than seven levels of products. Based on a beta test survey, testers were satisfied on its usability and functionality of this multi-level visualization tool. However, this approach does not convert graphical files into other widely used formats, and it does not offer many options to allow users to customize the predicted view.

In the last phase, computing methods were implemented to improve the prediction performance, and a simulation study was conducted to reduce the queuing interference. Based on the system usage statistics, we realized the importance of improving the performance on predicting multi-level pathways for a web-based application. First, performance tuning and configuration optimization were conducted for the Tomcat web server and the MySQL database server. Second, Java multi-thread programming was used and the optimal multi-thread configuration was determined to speed up each prediction cycle. Last, public submission time stamps were used to simulate the queuing interference and a second mirror server was adopted to reduce this interference. Overall, the prediction speed was doubled, and queuing interference was reduced by over 85%. In a beta test, testers agreed that the prediction speed is fast enough and the server was responsive when they used it. However, the current system is not able to divide its computing load to all available resource evenly and uniformly, thus further improvements are needed.

This study concluded that the new multi-level visualization improved the UM-PPS by displaying common intermediates and cleavage products in a tree-like structure. The new system fully supports existing metabolic logic entries, and it can produce a multi-level prediction within an acceptable time frame. Beta testers agreed that they can

now view prediction alternatives much more easily. They were also satisfied on its accessibility and stability. Overall, the improved UM-PPS displays better graphical results and predicts biodegradation pathway in a timely manner.

FUTURE WORK

The new version UM-PPS holds great promise for a better visualization of biodegradation prediction results and a faster prediction system. However, due to the complexity of biodegradation pathways, multi-level visualization alone is not enough to understand the prediction results thoroughly. Based on the current hierarchical output, additional studies may give us further insight into the prediction results. The current computing infrastructure is not most optimized, and additional optimization approaches may feather improve the prediction performance. This study could be expanded in the following areas.

First, the system will need better interoperability, zoom ability, and interactive features. Currently, the system only accepts submissions written in SMILES strings and outputs prediction results on a web page. Supporting other widely used file formats and implementing Import/Export features will allow users to better use UM-PPS with other software. Providing web service interfaces, such as SOAP [59] or REST [73], will allow software developers to integrate UM-PPS with other pipeline systems. The system can display multiple levels of predictions as a DAG that may contain over one hundred nodes and edges. When the prediction depth further increases, adding Expand/Collapse features will allow users to better understand the hierarchical structure from the overcrowd prediction results. The system displays all plausible predictions on one web page; adding a filtering feature will allow users to simplify the view by only showing selected pathways.

Second, it will be useful to display thermodynamics information and overall likelihood of a predicted pathway to suggest the best choice. In published pure culture studies, microbes tend to gain the maximal energy during the biodegradation process. Under this observation, thermodynamically efficient pathways may be more likely to occur than other alternatives. Recent studies also proposed methods to calculate free energy of chemical reactions [74]. Based on these studies, methods can be developed to

measure the energy efficiency by adding ATP equivalents produced over the entire pathway. Such information may guide users to evaluate the predicted pathways from a thermodynamics perspective. In a previous paper, we described super rules that consist of a group of contiguous rules and form small pathways of their own [2]. Super rules inherit likelihood values from their children rules and guide users faster to right choices. If we use a similar approach to assign overall likelihood values for all contiguous rules of a predicted pathway, it may allow us to better rate candidate pathways. But we have to carefully evaluate the risk of over-prediction before we add a new rating measurement.

Third, system usage estimation and queuing interference simulation need to be improved based a larger dataset. Although the usage of UM-PPS has been in a steady trend since 2008, the system usage may change in the future. The current computing resources were allocated based on the estimated system usage from archived submissions of 2009, and the improvements on queuing interference were confirmed by the dataset of 2010. But over estimating of system usage was also seen during to the difference between the older systems and the multi-level system. Since the available dataset of multi-level system is limited, it is impossible to accurately forecast the system usage and queuing interference in this study. When the use of multi-level UM-PPS increases, a larger dataset will permit better estimating of system usage and simulating of queuing interference. Therefore, the system capability can be updated to meet the future demands.

Last, server load balancing will further improve the computing performance. Currently the UM-PPS uses two sets of web servers; one is used as the primary server and another is used as a mirror server that processes additional submissions when the primary server is busy. However, this approach is not sufficient to maximally utilize a multi-server system's resource. First, it lacks a load balancing strategy that allocates submissions to a server with the least number of submissions in queue. Second, it lacks a strategy that divides a prediction task between two servers. These two deficiencies may waste system resources when uneven load balancing happens. With the increasing usage of the UM-PPS, such strategies will be highly desirable in the future.

The study presented in this dissertation contributed a major improvement in the UM-PPS visualization and performance for a better prediction of biodegradation pathways.

Chapter VII

REFERENCES

[1] Gao, J., Ellis, L.B.M., and Wackett, L.P. (2009). The University of Minnesota Biocatalysis/Biodegradation Database: improving public access. *Nucleic Acids Res.*, 38, D488-D491.

[2] Ellis, L.B.M., Gao, J., Fenner, K., and Wackett, LP. (2008). The University of Minnesota pathway prediction system: predicting metabolic logic. *Nucleic Acids Res.*, 36, W427-W432.

[3] Wuensch, K.L. (2005, October 4). *What is a Likert Scale?* Retrieved October 7, 2010, from http://core.ecu.edu/psyc/wuenschk/StatHelp/Likert.htm

[4] Hou, B.K., Ellis, L.B.M., and Wackett, L.P. (2004). Encoding Microbial Metabolic Logic: Predicting Biodegradation. *Nucleic Acids Res.*, 31, 261-272.

[5] Weininger, D. (1988). SMILES: a Chemical Language for Information Systems. 1. Introduction to Methodology and Encoding Rules. *J. Chem. Inf. Comput. Sci.*, 28, 31-36.

[6] Fenner, K., Gao, J., Kramer, S., Ellis, L.B.M., Wackett, L.P. (2008). Data-driven extraction of relative reasoning rules to limit combinatorial explosion in biodegradation pathway prediction. *Bioinformatics*, 24, 2079 - 2085.

[7] Long, A. (2002). Rule based prioritisation of metabolites. Some recent developments in METEOR. *Drug Metab. Rev.*, 34 (Suppl. 1), 71.

[8] Dimitrov, S., Nedelcheva, D., Dimitrova, N., and Mekenyan, O. (2010). Development of a biodegradation model for the prediction of metabolites in soil. *Sci. Total Environ.*, 408(18), 3811-3816.

[9] Moriya, Y., Shigemizu, D., Hattori, M., Tokimatsu, T., Kotera, M., Goto, S., and Kanehisa, M. (2010). PathPred: an enzyme-catalyzed metabolic pathway prediction server. *Nucleic Acids Res*., 38, W138-W143.

[10] Muir, D.C., Howard, P.H. (2006). Are there other persistent organic pollutants? A challenge for environmental chemists. *Environ. Sci. Technol.*, 40, 7157–7166.

[11] Theocharidis, A., van Dongen, S., Enright, A.J., and Freeman, T.C. (2009). Network Visualisation and Analysis of Gene Expression Data using BioLayout Express3D. *Nature Protoc.*, 4, 10:1535-50.

[12] Cline, M.S., et al. (2007). Integration of biological networks and gene expression data using Cytoscape. *Nat Protoc.*, 2(10):2366-82.

[13] Ellson, J., Gansner, E., Koutsofios, L., North, S.C., Woodhull, G. (2002). Graphviz - Open source graph drawing tools. *In: Mutzel, P., Junger, M., Leipert, S. Lecture Notes in Computer Science*., Berlin, Germany: Sprintger-Verlag, 2265, 483-484.

[14] Hooper, S.D., Bork. P. (2005). Medusa: a simple tool for interaction graph analysis. *Bioinformatics*, 21:4432–4433.

[15] Köhler, J., Baumbach, J., Taubert, J., Specht, M., Skusa, A., Rüegg, A., Rawlings, C., Verrier, P., Philippi, S. (2006). Graph-based analysis and visualization of experimental results with ONDEX. *Bioinformatics*, (11):1383-90.

[16] Breitkreutz, B.J., Stark, C., and Tyers, M. (2003). Osprey: a network visualization system. *Genome Biol.*, 4:R22.

[17] Batagelj, V., Mrvar, A. (2003). Pajek - Analysis and Visualization of Large Networks. In M. M. Jünger, *Graph Drawing Software* (pp. 77-103). Berlin: Springer.

[18] Demir, E., Babur, O., Dogrusoz, U., Gursoy, A., Nisanci, G., Cetin-Atalay, R., Ozturk, M. (2002). PATIKA: an integrated visual environment for collaborative construction and analysis of cellular pathways. *Bioinformatics*, 18(7):996-1003.

[19] Orlev, N., Shamir, R., and Shiloh, Y. (2003). PIVOT: Protein Interacions VisualizatiOn Tool. *Bioinformatics*, 20(3):424-425.

[20] Iragne, F., Nikolski, M., Mathieu, B., Auber, D., Sherman, D. (2005). ProViz: protein interaction visualization and exploration. *Bioinformatics*, 21(2):272-4.

[21] Hucka, M., Finney, A., Sauro, H.M., Bolouri, H., Doyle, J.C., Kitano, H., et al. (2003). The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, 19(4): 524–31.

[22] yWorks Company. (n.d.). *yFiles - Java Graph Layout and Visualization Library.* Retrieved November 30, 2010, from http://www.yworks.com/en/products_yfiles_about.html

[23] ChemAxon Company. (n.d.). *ChemAxon Metabolizer*. Retrieved November 30, 2010, from http://www.chemaxon.com/jchem/doc/user/metabolizer.html

[24] Csizmadia, F. (2000). JChem: Java applets and modules supporting chemical database handling from web browsers. *J. Chem. Inf. Comput. Sci.*, 40, 323–324.

[25] Firuziaan, M., Nommensen, O. (2002). *Parallel Processing via MPI & OpenMP*. Linux Enterprise.

[26] Carpente, B., Fox, G., Ko, S., Lim, S. (1999). *mpiJava 1.2: API Specification.* Retrieved November 30, 2010, from http://www.hpjava.org/reports/mpiJava-spec/mpiJava-spec/mpiJava-spec.html

[27] Carpenter, B., Getov, V., Judd, G., Skjellum, T., Fox. G. (2000). MPJ: MPI-like Message Passing for Java. *Concurrency: Practice and Experience*, 12(11):1019–1038.

[28] Genaud, S., Jeannot, E., Rattanapoka, C. (2009). Fault management in P2P-MPI. *International Journal of Parallel Programming*, 37(5): 433-461.

[29] Oracle Company. (n.d.). *Java Package java.util.concurrent.* Retrieved November 30, 2010, from http://download.oracle.com/javase/6/docs/api/java/util/concurrent/package-summary.html

[30] Jonathan, C.R. (2006). *Message passing interface vs. multi-threading: which parallelization technique is more efficient?* Retrieved November 30, 2010, from http://computing.ornl.gov/internships/rams/archive/rams06/websites/j_rann/images/Presentation.pdf

[31] Bieri, T., et al. (2006). WormBase: new content and better access . *Nucleic Acids Res.*, 35:D506-10.

[32] Roven, C., Bussemaker, H.J. (2003). REDUCE: an online tool for inferring cis-regulatory elements and transcriptional module activities from microarray data. *Nucleic Acids Res.*, 31(13): 3487-3490.

[33] Wang, N., Sherwood, A.R., Kurihara, A., Conklin, K.Y., Sauvage, T., Presting, G.G. (2009). The Hawaiian Algal Database: a laboratory LIMS and online resource for biodiversity data. *BMC Plant Biol.*, 9: 117.

[34] Chang, Y., Lin, Y., Ting, Y. (2006). Caching Personalized and Database-related Dynamic Web Pages. *IWNAS '06 Proceedings of the 2006 International Workshop on Networking, Architecture, and Storages.*

[35] *Java SE Documentation at a Glance*. (2010). Retrieved October 7, 2010, from http://www.oracle.com/technetwork/java/javase/documentation/index.html

[36] *Firefox 3.6 supports JavaScript 1.8.2*. (2009, June 22). Retrieved October 7, 2010, from https://developer.mozilla.org/en/firefox_3.6_for_developers#JavaScript

[37] *Java Applets home page*. (2010). Retrieved October 7, 2010, from http://java.sun.com/applets/

[38] *PNG (Portable Network Graphics) Home Site*. (2010, May 29). Retrieved October 7, 2010, from http://www.libpng.org/pub/png/

[39] *MySQL :: The world's most popular open source database*. (2010). Retrieved October 7, 2010, from http://www.mysql.com/

[40] *The Perl Programming Language*. (2010). Retrieved October 7, 2010, from http://www.perl.org/

[41] *The Apache HTTP Server Project*. (2010, September 21). Retrieved October 7, 2010, from http://httpd.apache.org/

[42] *Apache Tomcat home page*. (2010, August 11). Retrieved October 7, 2010, from http://tomcat.apache.org/

[43] *JavaServer Pages Technology*. (2010). Retrieved October 7, 2010, from http://java.sun.com/products/jsp/

[44] *Java Servlet Technology*. (2010). Retrieved October 7, 2010, from http://www.oracle.com/technetwork/java/index-jsp-135475.html

[45] Garrett, J.J. (2005, February 18). *Ajax: A New Approach to Web Applications.* Retrieved November 30, 2010, from http://www.adaptivepath.com/ideas/essays/archives/000385.php

[46] *Objects, Images, and Applets in HTML documents*. (2010). Retrieved October 7, 2010, from http://www.w3.org/TR/REC-html40/struct/objects.html

[47] Kanehisa, M., and Goto, S. (2000). KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic Acids Res.*, 28, 27-30.

[48] *Internet Explorer 8 home page*. (2010). Retrieved October 7, 2010

[49] *Mazilla Firefox web browsr home page*. (2010). Retrieved October 7, 2010, from http://www.mozilla.com/en-US/

[50] *Google Chrome home page.* (2010). Retrieved October 11, 2010, from http://www.google.com/chrome

[51] *Opera brower home page*. (2010). Retrieved October 7, 2010, from http://www.opera.com/

[52] *Apple Safari browser home page*. (2010). Retrieved October 7, 2010, from http://www.apple.com/safari/

[53] *Ubuntu home page*. (2010). Retrieved October 7, 2010, from http://www.ubuntu.com/

[54] *Apple Mac OS X Snow Leopard home page*. (2010). Retrieved October 7, 2010, from http://www.apple.com/macosx/

[55] *SDF Toolkit.* (2010, Januray 14). Retrieved October 7, 2010, from http://cactus.nci.nih.gov/SDF_toolkit/

[56] *Media:MDL CTfile Formats.* (2010, June). Retrieved October 7, 2010, from http://www.mdl.com/downloads/public/ctfile/ctfile.pdf

[57] *MDL CTfile Formats White Paper*. (2010). Retrieved October 7, 2010, from http://www.mdl.com/solutions/white_papers/ctfile_formats.jsp

[58] Shannon, P., Markiel, A., Ozier, O., Baliga, N.S., Wang, J.T., Ramage, D., Amin, N., Schwikowski, B., Ideker, T. (2003, November). Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.*, 13(11), 2498-504.

[59] *SOAP Version 1.2 specification.* (2007, April 27). Retrieved October 7, 2010, from http://www.w3.org/TR/soap12-part1/#intro

[60] *W3C SVG Working Group*. (2010). Retrieved October 7, 2010, from http://www.w3.org/Graphics/SVG/

[61] De Lorenzo V. (2008). Systems biology approaches to bioremediation. *Curr. Opin. Biotechnol.*, 19,579-589.

[62] Harrell, F.E. (2009). *R Package Hmisc.* Retrieved February 10, 2010, from R-Project: http://cran.r-project.org/web/packages/Hmisc/index.html

[63] *MySQL Administrator Best Practices.* (n.d.). Retrieved January 13, 2010, from http://dev.mysql.com/tech-resources/articles/mysql-administrator-best-practices.html

[64] Rick James. (2009, December 10). *Finding resources needed for sleeping threads.* Retrieved January 13, 2010, from http://forums.mysql.com/read.php?20,294895,295290#msg-295290

[65] *MySQL connection was unexpectedly lost.* (n.d.). Retrieved January 18, 2010, from http://forums.mysql.com/read.php?39,180347,227578

[66] Becker, R.A., Chambers, J.M., and Wilks, A.R. (1988). *The New S Language.* Wadsworth & Brooks/Cole.

[67] *A Standard for Robot Exclusion.* (n.d.). Retrieved November 30, 2010, from http://www.robotstxt.org/orig.html

[68] *Java Tuning White Paper*. (n.d.). Retrieved January 18, 2010, from Sun Developer Network: http://java.sun.com/performance/reference/whitepapers/tuning.html

[69] Shirazi, J. (2003). *Java performance tuning.* CA: O'Reilly Media.

[70] Wang,Y., Xiao,J., Suzek,T.O., Zhang,J., Wang,J. and Bryant,S.H. (2009). PubChem: a public information system for analyzing bioactivities of small molecules. *Nucleic Acids Res.*, 37, W623–W633.

[71] *GOOGLE BOT IP 66.249.67.* to 66.249.71.* .* (2010, February 10). Retrieved October 25, 2010, from Design Code Execute : http://www.designcodeexecute.com/2010/02/10/google-bot-ip-66-249-67-to-66-249-71-and-msn-bing-bot-crawls/

[72] von Ahn, L., Maurer, B., McMillen, C., Abraham, D., and Blum, M. (2008). reCAPTCHA: Human-Based Character Recognition via Web Security Measures. *Science*, 1465-1468.

[73] Elkstein, M. (n.d.). *What is REST?* Retrieved November 30, 2010, from http://rest.elkstein.org/2008/02/what-is-rest.html

[74] Jankowski, M.D., Henry, C.S., Broadbelt, L.J., and Hatzimanikatis, V. (2008). Group Contribution Method for Thermodynamic Analysis of Complex Metabolic Networks. *Biophys. J.*, (95) 3, 1487-1499.

Appendix A

UM-PPS BETA TEST SURVEY

SURVEY DESCRIPTION

The multi-level UM-PPS (version 2.0) has been under development since August 2009. Before its public release, we conducted a beta test to collect users' comments. The survey was developed by Dr. Lynda Ellis and Junfeng Gao. We recruited twelve volunteer UM-BBD users as beta testers. They come from: Canada, China (2), Germany, India, Iran, Japan, Netherlands, Switzerland (2), and USA (3). They have many different titles: Manager/Director (3), Faculty (1), Scientist (4), Research Fellow (1), Postdoctoral researcher (2), Student (2). They work at academic, industrial, and government organizations and institutions, and represent all types of UM-PPS users.

The beta test ran during the month of May 2010. Beta testers were required to login the test multi-level UM-PPS by using assigned username and password. They must follow the following instructions:

1. Read "About PPS" document.
   http://umbbd.msi.umn.edu/predict/aboutPPS.html
   http://umbbd.msi.umn.edu/predict/PPStutorial.pdf
2. Enter at least 15 compounds into the system (more if possible)
3. Optionally enter some of these compounds into the older system
4. Print at least 5 of the version 2.0 predictions to test printing capabilities
5. Complete an online survey when you are done with your test, no later than Monday, May 31, 2010.

We sent beta testers a link to the survey in May 2010, and we requested beta testers to take the survey only after they have completed the test. To catch any incorrect predictions, we cached all predictions entered during the time of the beta test. This allowed users to send us a link to a prediction if they have questions. Dr. Lynda Ellis used her email, lynda@umn.edu, to receive beta users' questions or comments.

SURVEY RESULTS

The beta test of multi-level UM-PPS received seven responses in June 2010. The results first presented the client environment, the frequency of using UM-BBD, the preference of between two systems, and the ranking of visualization options.

1. Operating systems used:
   6 Windows and 1 Macintosh
2. Web browsers used:
   6 Internet Explorer, 2 Firefox, and 1 Chrome
3. Printer used to print predictions:
   5 Monochrome and 2 Color
4. How often have you used the UM-BBD in the past?
   1 Daily, 2 Weekly, and 4 Monthly
5. What version of the UM-PPS would you prefer to use in the future:
   7 Version 2.0
6. Rank changeable options (most useful, first):
   Aerobic/all; number of levels; number of products on a level; minimum number of carbons to see; compound box size.

The results then presented numeric answers to the following questions. We used 5-point Likert scale measurement (5 = very much agree, 4 = agree, 3 = neutral, 2 = disagree, 1 = very much disagree), and used the mean of all seven responses.

7. Overall, the UM-PPS 2.0 is easy to use: 4.3
8. Overall, my experience using the UM-PPS 2.0 was satisfying: 4.3
9. Overall, I am satisfied in how UM-PPS 2.0 predictions look when they are printed: 3.6
10. Compounds on the screen are easy to see: 4.6
11. Prediction layouts (arrows and compound squares) are easy to understand: 4.6
12. Removal of duplicate products is important to me: 4.0
13. Indication of cleavage products is important to me: 4.4
14. The speed of a prediction is fast enough: 4.4
15. The system was responsive every time I tried it: 4.0